(1)

MULTIPLE MODEL ADAPTIVE ESTIMATION TECHNIQUES FOR ADAPTIVE MODEL-BASED ROBOT CONTROL

THESIS

Samuel John Sablan
Capt, USAF

AFIT/GE/ENG/89D-45

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY
AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

AFIT/GE/ENG/89D-45

MULTIPLE MODEL ADAPTIVE ESTIMATION TECHNIQUES FOR ADAPTIVE MODEL-BASED ROBOT CONTROL

THESIS

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air University
In Partial Fulfillment of the
Requirements for the Degree of
Master of Science in Electrical Engineering

Samuel John Sablan, B.S.E.E.
Capt, USAF

December, 1989

Approved for public release; distribution unlimited

Preface

The accomplishments of this research would not have been possible without the support and guidance of my thesis committee. I would like to give my sincere thanks to my advisor, Capt Michael B. Leahy Jr. His constant support and assistance came at crucial times during my research. Without his encouragement, I would have not been able to finish. I would also like to express my deepest appreciation to Dr. Peter Maybeck whose expertise in stochastic estimation and adaptation schemes provided the insight to solve this difficult problem. And I would like to thank Dr Gary Lamont for his comments and suggestions on the document itself.

In addition to my committee members, I would like to thank Mr. Dan Zambon for his invaluable assistance with the computers used to accomplish this research. Also, I would like to thank Captains Vern Milholen and Mark Johnson for their encouragement and companionship over many late nights in the Lab. Finally to my wife, Susun, I want her to know that without her love and support all this would have been meaningless.

Samuel John Sablan

Table of Contents

List of Figures

Figure Page

Figure Page

Figurɩ Page

Figure Page

Figure Page

List of Tables

Table Page

AFIT/GE/ENG/89D-45

Abstract

The use of robotic manipulatuis for future Air Force applications will require a manipulator capable of emulating the performance of the human arm. To emulate human arm motion, a robot must be capable of adapting quickly and accurately to changes in the environment while maintaining accurate high speed tracking performance. One approach to adaptive robotic control is the use of Multiple Model Adaptive Estimation (MMAE) techniques within a model-based control structure. The MMAE techniques employ a bank of Kalman filters whose models are based on different assumed values of the uncertain paramcters. Using this bank of filters, the MMAE provides an estimate of the uncertain parameters. A previous development used a closed-loop form of MMAE with a model-based controller and was called Multiple Model-Based Control (MMBC). Further analysis of the MMBC showed it has limited applications to manipulators whose dynamics and tracking performance depend heavily on the payload. This is not the case for the PUMA-560 manipulator. As a result, a new form of adaptive model-based control called Open-Loop Multiple Model-Based Control (OL/MMBC) was developed. The OL/MMBC combines a model-based controller with a MMAE algorithm whose filters are based on an open-loop linearized perturbation model. The OL/MMBC was simulated and experimentally evaluated on a PUMA-560 manipulator. The OL/MMBC algorithm adapted quickly to uncertain payloads and provide payload estimates which resulted in tracking performance significantly better than a model-based controller without knowledge of the payload. Additionally, the tracking performance was comparable to a model-based controller artificially informed of the payload. The experimental evaluations validated the simulation results and show the potential of the OL/MMBC for possible use in future Air Force robotic applications.

MULTIPLE MODEL ADAPTIVE ESTIMATION TECHNIQUES FOR ADAPTIVE MODEL-BASED ROBOT CONTROL

I. Introduction

1.1 Motivation

The Air Force intends to use robotic manipulators to replace humans operating in hazardous conditions. Therefore, future robotics applications, such as robotic telepresence or flightline refueling, will require a manipulator capable of emulating the performance of the human arm. To emulate human arm motion, a robot must be capable of adapting quickly and accurately to changes in the environment while maintaining accurate high speed tracking performance. Advances in robotic control system design are necessary to provide human-like performance.

1.2 Research Objective

Current research at the Air Force Institute of Technology Robotic Systems Laboratory is centered on developing and evaluating various techniques that can adapt quickly to payload variations within the framework of a model-based control structure [11,15,30,31]. An adaptive model-based controller was recently developed that uses a stochastic multiple model adaptive estimation (MMAE) scheme to estimate the payload mass and update the feedforward compensation torques during manipulator motion [30]. The objective of this research effort is to continue the development and evaluation of MMAE techniques applied to adaptive model-based robot control.

1.3 Problem Statement

The control algorithm is the means by which the robot is commanded to follow a desired trajectory. The conventional control algorithms used in industry today attempt to maintain the desired trajectory by using either a Proportional Derivative (PD) or Proportional Integral Derivative (PID) feedback controller [6].

The PD or PID controllers feedback the measured errors in an attempt to drive the errors to zero. As long as these errors remain small, the PD or PID controller provides adequate performance. Unfortunately, as the speed of the trajectory increases or the configuration of the robot changes, the PD or PID controllers cannot maintain track along the desired trajectory. The main reason for poor tracking is that the PD and PID controllers were developed based on a simplified linear dynamics model that neglects the nonlinear dynamic effects. As speed increases, so do disturbances from dynamic effects, and the large disturbances result in poor tracking performance.

Therefore, an area of active research is the development of adaptive control algorithms that account for the nonlinear dynamic effects inherent in a robotic manipulator. A basic form of adaptive control is the model-based approach. In model-based control, the controller adapts to changes in robot configurations (e.g., speed and known payload variations). Simulation and experimental evaluations have demonstrated the potential for the model-based techniques to improve high speed tracking accuracy [1,10,33]. Unfortunately, the tracking performance degrades in the presence of uncertain payload parameters [1,11,12]. Adaptive control of robots is an area of active research [1,2,4,7,8,16,17,18,22,24,26,27,30]. One approach to restoring tracking accuracy has been to augment the model-based controller with an adaptation mechanism. Two current approaches to adaptive model-based control are based on Lyapunov theory and stochastic estimation and adaptation techniques [1,22,26,27,30]. In both of these approaches, an adaptive estimation algorithm is incorporated into the model-based control structure.

The approach based on Lyapunov theory guarantees that the controller will be stable and the steady state errors will asymptotically approach zero [1]. However, there is no way to predict how fast the errors will converge to zero. Human arm emulation demands quick convergence. An alternate approach is to use stochastic estimation and adaptation techniques. These techniques not only provide a fast means of parameter adaptation, but account for the numerous sources of noise and uncertainties in a real physical system.

In his thesis research, Tellman developed a form of adaptive model-based robot control based on stochastic estimation and adaptation techniques [30]. The basis of Tellman's adaptive algorithm was to use a Bayesian estimation approach that employs multiple Kalman filters to estimate parameters quickly and accurately in the presence of noise and uncertainty [30:Chapter 3]. This Bayesian estimation approach is known as Multiple Model Adaptive Estimation (MMAE) [20:129-136]. The MMAE developed by Tellman was a closed-loop form of the MMAE techniques called ΔMMAE and estimated the parameter variations (the ΔMMAE will be discussed later in Chapter 3). The ΔMMAE technique was incorporated into a model-based control structure to develop a new form of adaptive model-based control called Multiple Model-Based Control (MMBC). Tellman simulated but did not experimentally evaluate the MMBC. In order to test the true potential of any control algorithm, an experimental evaluation must be conducted. The experimental evaluation will not only validate the results obtained in simulation, but move one step closer to eventual implementation.

1.4 Approach

The first step in this research effort was to evaluate the existing MMBC algorithm to gain an understanding of how algorithm worked. During this evaluation, errors were discovered in some of the software programs used in simulation studies and were subsequently corrected. The corrections changed the dynamic model of the robot and required retuning the Kalman filter noises used in the ΔMMAE.

Rigorous attempts at retuning the filters were unsuccessful. The retuning process simulated the PUMA-560 manipulator carrying a payload of 4.0 Kg while not informing the model-based controller of the estimator's payload estimate. In this configuration, the retuning process attempted to achieve the best estimate of the 4.0 Kg payload variation between the uninformed model-based controller and the simulated manipulator payload. The inability to retune the filters prompted further analysis of the MMBC in a attempt to gain insights into a modification to allow for better tuning.

The linear perturbation models used to develop the Kalman filters of the ΔMMAE were based on perturbations about some nominally desired trajectory and torque. The nominally desired torque was assumed to be the feedforward torque of a model-based controller with payload knowledge. Therefore each Kalman filter was based on a different nominal torque, and any perturbation from the nominal torque was assumed to be handled by the PD feedback loop. This resulted in a closed-loop linearized plant matrix which had little dependence on the payload. This slight dependence on payload and the changes in the dynamics model made it difficult for the ΔMMAE to estimate payload. This prompted an evaluation of the commanded and feedforward torques of a single model-based controller that could be artificially informed of the payload.

The evaluations revealed that even without knowledge of the payload the overall commanded torque was similar to the feedforward torque of a model with payload informations. This is important because it indicates that the smallest perturbations would occur between the commanded torque and the nominal torque of the filter model whose assumed payload closely matches the actual unknown payload.

Using this information, an alternate design was developed that did not assume the perturbation torques of the linearized perturbation model were the PD feedback torques of a model-based controller. This alternate design uses the difference between the overall commanded torque applied to the arm and the nominally desired torques assumed by each filter to help separate the filter models and make payload estimation possible. This resulted in an alternate form of MMBC which is called Open-Loop MMBC (OL/MMBC). Therefore, OL/MMBC no longer estimates payload parameter variations but actual payload. The reason this algorithm is called an open-loop MMBC is the fact that the PD feedback loop was no longer used as part of the linearized plant matrix which opened the loop of the linearized perturbation model.

The first three links of a PUMA-560 manipulator were used as a case study to evaluate the OL/MMBC. After tuning the Kalman filter noises to achieve the best payload estimation performance, simulations were conducted for various payloads using a minimum-jerk trajectory profile that moved the PUMA-560 from (−50°, −135°, 135°) to (−50°, −135°, 135°) in 2.0 seconds. Following these simulations, the OL/MMBC was experimentally evaluated using the same trajectory profile and payloads of 0.0 and 3.0 Kg.

1.5 Accomplishments

Evaluations of the original MMBC algorithm demonstrated that it is limited to applications where both robot tracking performance and dynamics depended heavily on payload. Also, a new form of MMAE based robot control called open-loop MMBC applicable to situations where the tracking but not the dynamics depended heavily on payload was developed and analyzed.

The initial simulations of the OL/MMBC for a PUMA-560 manipulator showed its ability to adapt quickly to the unknown payload information. The tracking performance was significantly better than a model-based controller without knowledge of the payload and comparable to a model-based controller that was artificially informed of the payload. The experimental results validated the simulations and showed the potential of MMAE techniques for use in adaptive model-based robot control applications. Overall, the OL/MMBC has demonstrated the ability to adapt quickly to payload uncertainties and maintain tracking performance.

1.6 Organization

The remainder of the thesis is presented in five chapters. Chapter 2 presents a literature review of current adaptive robot control schemes with emphasis placed on adaptive model-based control approaches. Chapter 3 discusses the development of Tellman's MMBC algorithm, the errors discovered during the evaluation of the software code, and how this led to the development of a the modified MMBC algorithm. Chapter 4 presents a development of the open-loop MMBC algorithm. Chapter 5 presents the details of implementing the modified MMBC algorithm for a PUMA-560 manipulator, the simulation and experimental evaluation, and the results of those evaluations. Chapter 6 presents the conclusions and recommendations for further research.

II. Literature Review

2.1 Introduction

An area of active research is the development of an adaptive robot control algorithm. Various forms of adaptive robot control algorithms have been developed. These include model-referenced adaptive control (MRAC) [4,18,24], adaptive control using an autoregressive model [8], adaptive perturbation control [2,16,17,7], and adaptive model-based control [1,10,11,22,26,27,30,33].

Since this thesis effort is continuing the previous research that uses model-based adaptive control, the literature review will emphasize those algorithms based on model-based adaptive control. A short summary of the other approaches and review of model-based adaptive control follows. A more detailed review of the other approaches can be found in [30].

2.2 Model Reference Adaptive Control (MRAC)

The basic principle behind model-referenced adaptive control (MRAC) is to select a reference model and adaptation algorithm which modifies the feedback gains of the actual robot controller [6:244–245]. A general block diagram of the MRAC is shown in Figure 2.1.

A reference input is applied to both the robot and the reference model. The error between the outputs of the robot ($\mathbf{x}$) and reference model ($\mathbf{y}$) drive the adaptation algorithm which modifies the feedback gains. This modification in the gains attempts to asymptotically drive the errors to zero. Therefore, the goal is to have the the robot outputs track the reference model outputs.

For example, Dubowsky and DesForges [4] first proposed a simple MRAC for robot manipulators back in 1979. In their proposed MRAC algorithm they used a linear second-order time invariant reference model represented as [4,6]

$$a_i \ddot{y}_i(t) + b_i \dot{y}_i(t) + y_i(t) = r_i(t) \tag{2.1}$$

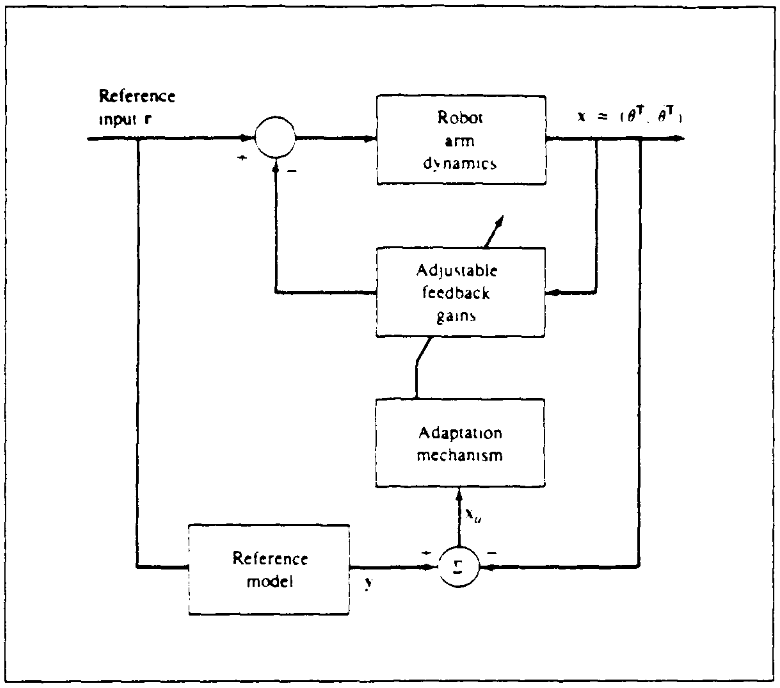


Figure 2.1. General Block Diagram for MRAC [6]

where a_i and b_i are associated with the *i*th joint of the manipulator and correspond to the natural frequency ω_{n_i} and damping ratio ζ_i by

$$a_i = \frac{1}{\omega_{n_i}^2} \quad and \quad b_i = \frac{2\zeta_i}{\omega_{n_i}} \tag{2.2}$$

The choice of the reference model is based on the assumption that the nonlinear coupling terms are negligible and the links are counterbalanced to minimize the effects of gravity. Thus the *i*th joint dynamic equations can be written as [4,6]

$$\alpha_i \ddot{x}_i(t) + \beta_i \dot{x}_i(t) + x_i(t) = r_i(t) \tag{2.3}$$

The system parameters α_i and β_i are assumed to vary slowly with time and are adjusted using a steepest decent method to minimize a quadratic function of the errors between the reference model and the robot outputs. The adjusted system parameters are then used to modify the feedback gains.

The assumptions about nonlinear coupling made it possible for Dubowsky and DesForges to assume a linear second-order model for the system dynamics. However, this assumption would only be valid for very slow trajectories in which the nonlinear dynamic effects do not come into play. Unfortunately, future robotic applications must be capable of tracking fast trajectories.

2.3 Adaptive Control Using an Autoregressive Model

An adaptive control using an autoregressive model proposed by Koivo and Guo [8] relates the input-output data from the manipulator. In their control algorithm they assume the nonlinear coupling dynamic effects are negligible like Dubowsky and DesForges. Furthermore the autoregressive model is of a stochastic nature using a modeling error term assumed to be zero-mean white Gaussian noise and independent of the inputs and outputs.

The principle behind this form of adaptive control is to use an autoregressive model to get a best fit of the input-output data from the manipulator. Then, using a recursive least-squares identification scheme, the coefficients of the autoregressive model are estimated and used in an optimal control law to minimize the performance criterion of tracking a desired trajectory. A block diagram of this control algorithm is shown in Figure 2.2. The desired trajectory is denoted as $\mathbf{y}^d$ and the torques are the control inputs $\mathbf{u}$. The outputs of the manipulator $(\mathbf{y})$ are the actual joint position and velocities.

2.4 Adaptive Perturbation Control

In adaptive perturbation control the nonlinear coupled dynamics of the robot manipulator are linearized about some nominal trajectory using a first-order Taylor series approximation. This approximation leads to a set of linearized perturbation equations about the nominal trajectory.

Lee and Chung [16,17] and Guo and Angeles [7] developed adaptive control algorithms based on these linearized perturbation equations. The basic principle behind their algorithms is to break the controi system into a feedforward and feedback component. The feedforward component computes a nominal torque based on the nonlinear robot dynam-

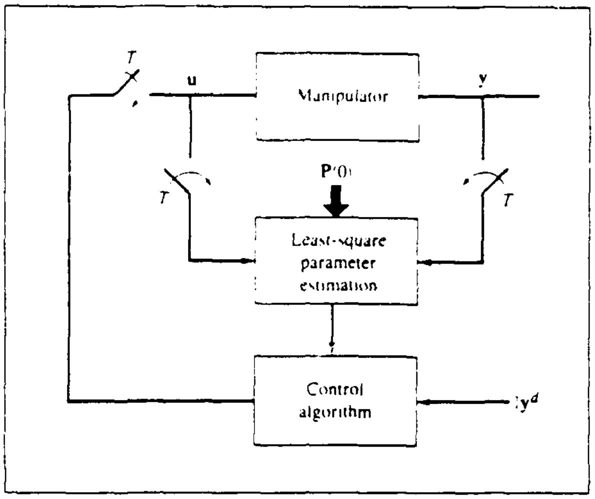


Figure 2.2. Adaptive Control using Autoregressive Model [6]

ics model. Then based on the linearized perturbation equations, a feedback perturbation torque is computed using a recursive least-squares to identify the system parameters in the perturbation equations. These estimates are used to provided the optimal feedback torque. Therefore, the total torques applied to the manipulator consist of the nominal torque plus the perturbation torque computed from the optimal control law. A block diagram of the adaptive perturbation control algorithm used by Lee and Chung is shown in Figure 2.3 and a good discussion of the alogorithm can be found in [6,16].

2.5 Adaptive Model-Based Control

The basic model-based approach adapts to manipulator configuration changes, and simulation and experimental evaluations have demonstrated the potential for the model-based techniques to improve high speed tracking accuracy [1,10,33]. However, the tracking performance degrades in the presence of uncertain payload parameters [1,11]. Therefore, an approach has been to augment the model-based controller with an adaptation mechanism [1,22,26,27,30].

Two current approaches to adaptive model-based control are based on Lyapunov theory, and stochastic estimation and adaptation techniques. Background on model-based robot control and review of Lyapunov adaptive approaches are presented in the following

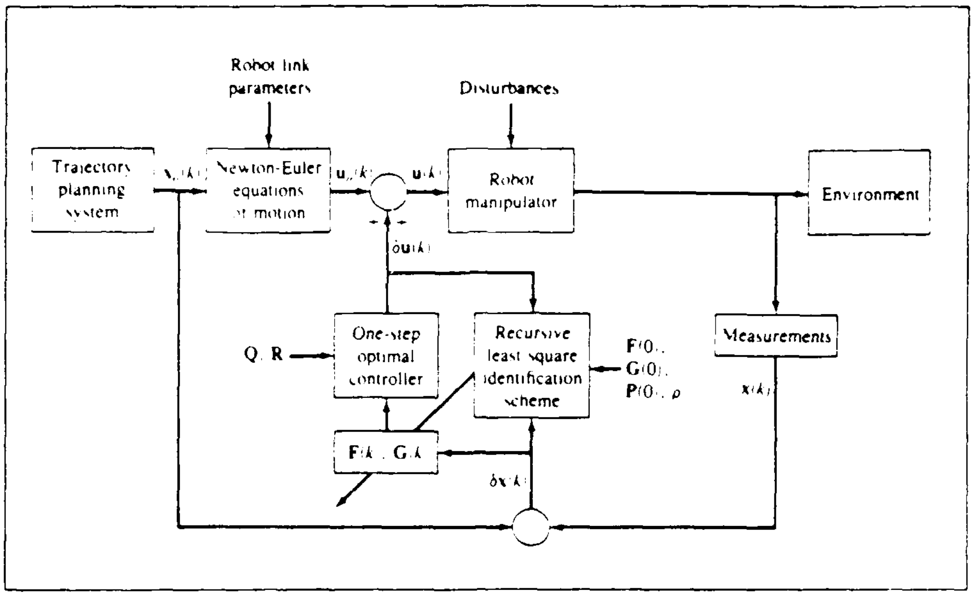


Figure 2.3. Adaptive Perturbation Control Algorithm [6]

sections.

2.5.1 Model-Based Control Background In order to use model-based control, a model of the dynamics of the robot manipulator must be developed. The equations of motion for a rigid robot manipulator model can be written in vector form as [5,10,11,12]:

$$\mathbf{N}\boldsymbol{\tau}_m(t) = [\mathbf{D}(\mathbf{q},\mathbf{a}) + \mathbf{N}^2\mathbf{J}_m]\ddot{\mathbf{q}} + \mathbf{h}(\dot{\mathbf{q}},\mathbf{q},\mathbf{a}) + \mathbf{N}^2\mathbf{B}_m\dot{\mathbf{q}} + \boldsymbol{\tau}_s + \mathbf{g}(\mathbf{q},\mathbf{a}) \qquad (2.4)$$

where:

- n = number of degrees of freedom of the robot manipulator.
- $\mathbf{q}, \dot{\mathbf{q}}, \ddot{\mathbf{q}}$ = n-dimensional vectors of joint positions, velocities, and accelerations.
- $\mathbf{a}$ = 10-dimensional vector representing the mass, mass centroid, and radii of gyration of the payload.
- $\mathbf{N}$ = $n \times n$ diagonal matrix of gear ratios $\left(\frac{motor\ velocity}{link\ velocity}\right)$.
- $\boldsymbol{\tau}_m(t)$ = an n-dimensional vector of joint motor torques.
- $\mathbf{D}(\mathbf{q},\mathbf{a})$ = an $n \times n$ matrix of manipulator inertia terms.

- $\mathbf{J}_m$ = an $n \times n$ diagonal matrix of actuator inertia terms.
- $\mathbf{h}(\dot{\mathbf{q}}, \mathbf{q}, \mathbf{a})$ = an n-dimensional vector of centripetal and Coriolis torques.
- $\boldsymbol{\tau}_s$ = an n-dimensional vector of static friction torques.
- $\mathbf{B}_m$ is an n-dimensional vector of damping coefficients.
- $\mathbf{g}(\mathbf{q}, \mathbf{a})$ = an n-dimensional vector of gravity torques.

The intent of model-based control is to develop a control law of the form [10,11,12]:

$$\mathbf{N}\boldsymbol{\tau}_m(t) = \boldsymbol{\tau}_{ff} + \boldsymbol{\tau}_{fb} \tag{2.5}$$

where:

- $\boldsymbol{\tau}_{ff}$ = feedforward compensation torques.
- $\boldsymbol{\tau}_{fb}$ = feedback compensation torques.

Therefore, letting the right hand side terms of Equation (2.5) become [10,11,12]:

$$\boldsymbol{\tau}_{ff} = [\hat{\mathbf{D}}(\mathbf{q}, \mathbf{a}) + \mathbf{N}^2\mathbf{J}_m](\mathbf{q}, \mathbf{a})\ddot{\mathbf{q}} + \hat{\mathbf{h}}(\dot{\mathbf{q}}, \mathbf{q}, \mathbf{a}) + \mathbf{N}^2\mathbf{B}_m\dot{\mathbf{q}} + \boldsymbol{\tau}_s + \hat{\mathbf{g}}(\mathbf{q}, \mathbf{a}) \tag{2.6}$$

$$\boldsymbol{\tau}_{fb} = \mathbf{K}_v\,\dot{\mathbf{e}} + \mathbf{K}_p\,\mathbf{e} \tag{2.7}$$

where:

- $\dot{\mathbf{e}} = \dot{\mathbf{q}}_d - \dot{\mathbf{q}}$
- $\mathbf{e} = \mathbf{q}_d - \mathbf{q}$
- $\mathbf{q}_d, \dot{\mathbf{q}}_d, \ddot{\mathbf{q}}_d$ = n-dimensional vectors of desired joint positions, velocities, and accelerations.
- $\mathbf{K}_v, \mathbf{K}_p$ = diagonal $n \times n$ matrices of velocity and position feedback loop gains.
- $(\hat{\ })$ = modeled values.

Substituting Equations (2.6) and (2.7) into Equation (2.5), and equating Equations (2.5) and (2.4) (assuming the modelled values equal the actual values) yields

$$\ddot{\mathbf{e}} + \mathbf{K}_v \dot{\mathbf{e}} + \mathbf{K}_p \mathbf{e} = 0 \tag{2.8}$$

Equation (2.8) is the "ideal" situation and allows for the appropriate K_v and K_p to reject the errors, **e**, uniformly over the entire trajectory of the manipulator. Since Equation (2.4) is only a model of the actual dynamics of the manipulator, there are some uncertainties in the dynamics model parameters. These uncertainties show up as disturbances to the model-based controller. If the disturbances are small, then the controller can maintain good high speed tracking performance . When the payload parameters **a** are not known *a priori*, the performance of the model-based controller degrades significantly [1,11,12]. Therefore, the model-based approach must be augmented with an adaptation algorithm.

2.5.2 Lyapunov Theory Adaptive Approach Craig *et al.* [1], Middleton and Goodwin [22], and Slotine and Li [26,27] have used Lyapunov theory to develop their adaptation algorithms. Lyapunov theory guarantees that the controller will be stable and the steady state errors will asymptotically approach zero.

Craig *et al.* proposed using tracking errors in the joint positions and velocities to drive the adaptive algorithm for estimating the uncertain parameters [1]. The basic adaptation law was given by [1:page 20]:

$$\dot{\hat{\mathcal{F}}} = \mathbf{\Gamma}^{-1}\mathbf{W}^T\hat{\mathbf{D}}^{-1}(\mathbf{q})\mathbf{E} \tag{2.9}$$

where:

- $\hat{\mathcal{F}}$ = the vector of estimated uncertain parameters.
- $\mathbf{\Gamma}$ = the diagonal scaling matrix.
- $\mathbf{W}^T$ = the matrix of errors associated with the uncertain parameters of the dynamics model.
- $\hat{\mathbf{D}}(\mathbf{q})$ = the estimated manipulator inertia matrix.

- $\mathbf{E}$ = the vector of joint position and velocity errors $(\mathbf{e}, \dot{\mathbf{e}})$.

By solving Equation (2.9), the update for the estimate of the unknown parameters is obtained. An inherent problem with this approach is the requirement for joint acceleration measurements. These measurements are required for the $\mathbf{W}^T$ term of Equation (2.9).

Craig claims that the need for good acceleration measurements is not required due to the integration performed in solving Equation (2.9) [1:page 21]. This would only be true for constant velocity trajectories or those trajectories for which the period of accelerations were short.

Unfortunately, most high speed trajectories of interest do not allow for these assumptions. Therefore, to improve the tracking performance of the manipulator, the $\mathbf{\Gamma}$ term must be adjusted experimentally to meet the desired performance [1:26]. By adjusting $\mathbf{\Gamma}$, a compromise between peak tracking errors and speed of the adaptive algorithm results.

Middleton and Goodwin proposed an adaptive algorithm nearly identical to Craig *et al.*, but their algorithm does not require acceleration measurements in the adaptation law [22]. However, both approaches require the inversion of the estimated manipulator inertia matrix. This could lead to problems because of computation time, and more importantly, the estimated inertia matrix can become singular (noninvertible) due to computer rounding errors.

To try and alleviate those problems, Slotine and Li developed an adaptive algorithm which did not require the inversion of the estimated inertia matrix or acceleration measurements. Their adaptation scheme was based on Lyapunov theory like [1,22]. Simulation results showed that the tracking errors were small; however, the convergence time was slow [26:57-8].

A drawback to the adaptive techniques based on Lyapunov theory is not being able to predict how fast the adaptation law will converge. Therefore, an alternate approach is the use of stochastic estimation and adaptation techniques. These techniques not only provide a fast means of parameter adaptation but account for the numerous sources of noise and uncertainties in a real physical system. Tellman [30] recently developed a form of adaptive model-based robot control based on stochastic estimation and adaption techniques. A review of that development follows.

2.6 Multiple Model Adaptive Estimation Development

The basis of Tellman's adaptive algorithm was to use a Bayesian estimation approach that employs multiple Kalman filters to estimate parameters quickly and accurately in the presence of noise and uncertainty [30:Chapter 3]. This Bayesian estimation approach is known as Multiple Model Adaptive Estimation (MMAE) [20:129–136]. A short summary of the MMAE algorithm follows. For a more detailed development see Maybeck [20:129–136].

Assume the system being considered can be modeled by a discrete linear stochastic state equation such that [20]:

$$\mathbf{x}(t_{i+1}) = \mathbf{\Phi}(t_{i+1}, t_i)\mathbf{x}(t_i) + \mathbf{B}_d(t_i)\mathbf{u}(t_i) + \mathbf{G}_d(t_i)\mathbf{w}_d(t_i) \tag{2.10}$$

and the associated discrete measurements can be modeled as [20]:

$$\mathbf{z}(t_i) = \mathbf{H}(t_i)\mathbf{x}(t_i) + \mathbf{v}(t_i) \tag{2.11}$$

where:

- $\mathbf{x}(t_i)$ = n-dimensional state vector stochastic process.
- $\mathbf{\Phi}(t_{i+1}, t_i) = n \times n$ state transition matrix.
- $\mathbf{u}(t_i)$ = r-dimensional known control input vector.
- $\mathbf{B}_d(t_i) = n \times r$ control input matrix.
- $\mathbf{w}_d(t_i)$ = s-dimensional white Gaussian dynamics noise vector stochastic process.
- $\mathbf{G}_d(t_i) = n \times s$ dynamics noise input matrix.
- $\mathbf{z}(t_i)$ = m-dimensional measurement vector stochastic process.
- $\mathbf{H}(t_i) = n \times m$ measurement matrix.
- $\mathbf{v}(t_i)$ = m-dimensional white Gaussian measurement noise vector stochastic process.

Furthermore, assume $\mathbf{w}_d(t_i)$ and $\mathbf{v}(t_i)$ are independent for all time with the following statistics:[20]

$$\begin{aligned}
E\{\mathbf{w}_d(t_i)\} &= \mathbf{0} \\
E\{\mathbf{w}_d(t_i)\mathbf{w}_d^T(t_j)\} &= \mathbf{Q}_d(t_i)\delta_{ij} \\
E\{\mathbf{v}(t_i)\} &= \mathbf{0} \\
E\{\mathbf{v}(t_i)\mathbf{v}^T(t_j)\} &= \mathbf{R}(t_i)\delta_{ij}
\end{aligned}$$

where:

- δ_{ij} = Kronecker delta function.
- $\mathbf{Q}_d(t_i)$ = covariance of the dynamics noise.
- $\mathbf{R}(t_i)$ = covariance of the measurement noise.

If $\mathbf{a}$ denotes a vector of uncertain parameters in the given assumed model that may affect $\mathbf{\Phi}$, $\mathbf{B}_d$, $\mathbf{H}$, $\mathbf{Q}_d$, and $\mathbf{R}$, then the purpose of the Bayesian estimator is to compute the conditional density function: [20]

$$f_{\mathbf{x}(t_i),\mathbf{a}|\mathbf{Z}(t_i)}(\boldsymbol{\xi},\boldsymbol{\alpha}|\mathbf{Z}_i) = f_{\mathbf{x}(t_i)|\mathbf{a},\mathbf{Z}(t_i)}(\boldsymbol{\xi}|\boldsymbol{\alpha},\mathbf{Z}_i)f_{\mathbf{a}|\mathbf{Z}(t_i)}(\boldsymbol{\alpha}|\mathbf{Z}_i) \tag{2.12}$$

The first density term on the right hand side of Equation (2.12), based on the assumed model, would be Gaussian with mean $\hat{\mathbf{x}}(t_i{}^+)$ and covariance $\mathbf{P}(t_i{}^+)$. The mean and covariance would be computed by a Kalman filter for each of the given values of $\mathbf{a}$.

The second density term on the right hand side of Equation (2.12) can be evaluated as: [20]

$$f_{\mathbf{a}|\mathbf{Z}(t_i)}(\boldsymbol{\alpha}|\mathbf{Z}_i) = \frac{f_{\mathbf{z}(t_i)|\mathbf{a},\mathbf{Z}(t_{i-1})}(\boldsymbol{\zeta}_i|\boldsymbol{\alpha},\mathbf{Z}_{i-1})f_{\mathbf{a}|\mathbf{Z}(t_{i-1})}(\boldsymbol{\alpha}|\mathbf{Z}_{i-1})}{\int_A f_{\mathbf{z}(t_i)|\mathbf{a},\mathbf{Z}(t_{i-1})}(\boldsymbol{\zeta}_i|\boldsymbol{\alpha},\mathbf{Z}_{i-1})f_{\mathbf{a}|\mathbf{Z}(t_{i-1})}(\boldsymbol{\alpha}|\mathbf{Z}_{i-1})d\boldsymbol{\alpha}} \tag{2.13}$$

Equation (2.13) could conceptually be solved recursively starting from an *a priori* density $f_{\mathbf{a}}(\boldsymbol{\alpha})$ since $f_{\mathbf{z}(t_i)|\mathbf{a},\mathbf{Z}(t_{i-1})}(\boldsymbol{\zeta}_i|\boldsymbol{\alpha},\mathbf{Z}_{i-1})$ is Gaussian with a mean $\mathbf{H}(t_i)\hat{\mathbf{x}}(t_i^-)$ and covariance

$[\mathbf{H}(t_i)\mathbf{P}(t_i^-)\mathbf{H}^T(t_i) + \mathbf{R}(t_i)]$. The mean and covariance are computed by a Kalman filter for each given value of the parameter vector. [20]

Then using the conditional mean, the estimate of $\mathbf{x}(t_i)$ becomes: [20]

$$E\{\mathbf{x}(t_i|\mathbf{Z}(t_i) = \mathbf{Z}_i\} = \int_A \left[\int_{-\infty}^{\infty} \xi f_{\mathbf{x}(t_i),\mathbf{a}|\mathbf{Z}(t_i)}(\boldsymbol{\xi}, \boldsymbol{\alpha}|\mathbf{Z}_i)\mathrm{d}\boldsymbol{\xi}\right] f_{\mathbf{a}|\mathbf{Z}(t_i)}(\boldsymbol{\alpha}|\mathbf{Z}_i)\mathrm{d}\boldsymbol{\alpha} \tag{2.14}$$

where the term in brackets on the right hand side of Equation (2.14) is the estimate of $\mathbf{x}(t_i)$ as computed by the Kalman filter based on the realization of the parameter vector. If a is continuous over the range A, then an infinite number of Kalman filters would be required to estimate $\mathbf{x}(t_i)$. This would not be feasible to implement. Therefore, the parameter space is discretized so only a finite number of Kalman filters are required.

Letting the hypothesis conditional probability be defined as: [20]

$$p_k(t_i) \triangleq prob[\mathbf{a} = \mathbf{a}_k|\mathbf{Z}(t_i) = \mathbf{Z}_i] \tag{2.15}$$

a development totally analogous to Equations (2.13) and (2.14) produces the following: [20]

$$p_k(t_i) = \frac{f_{\mathbf{z}(t_i)|\mathbf{a},\mathbf{Z}(t_{i-1})}(\mathbf{z}_i|\boldsymbol{\alpha}_k, \mathbf{Z}_{i-1})p_k(t_{i-1})}{\sum_{j=1}^{K} f_{\mathbf{z}(t_i)|\mathbf{a},\mathbf{Z}(t_{i-1})}(\mathbf{z}_i|\boldsymbol{\alpha}_j, \mathbf{Z}_{i-1})p_j(t_{i-1})} \tag{2.16}$$

$$\hat{\mathbf{x}}(t_i^+) = E[\mathbf{x}(t_i)|\mathbf{Z}(t_i) = \mathbf{Z}_i] = \sum_{k=1}^{K} \hat{\mathbf{x}}_k(t_i^+)p_k(t_i) \tag{2.17}$$

where $\hat{\mathbf{x}}_k(t_i^+)$ is the state estimate computed by a Kalman filter based on the parameter vector equal to $\mathbf{a}_k$. A block diagram of the MMAE algorithm is shown in Figure 2.4. The MMAE algorithm consists of a bank of K Kalman filters each based on a particular value ($\mathbf{a}_1$,$\mathbf{a}_2$,...,$\mathbf{a}_K$) of the parameter vector. At the ith sample time, the measurement $\mathbf{z}_i$ is passed to each of the filters. The residuals ($\mathbf{r}_1(t_i)$,$\mathbf{r}_2(t_i)$,...,$\mathbf{r}_K(t_i)$) generated by the K filters are passed to the hypothesis conditional probability as computed by Equation (2.16). These hypothesis conditional probabilities are used as weighting factors to generate $\hat{\mathbf{x}}(t_i^+)$.

To compute the hypothesis conditional probabilities of Equation (2.16) requires the evaluation of $f_{\mathbf{z}(t_i)|\mathbf{a},\mathbf{Z}(t_{i-1})}(\mathbf{z}_i|\boldsymbol{\alpha}_k, \mathbf{Z}_{i-1})$. This can be evaluated as: [20]

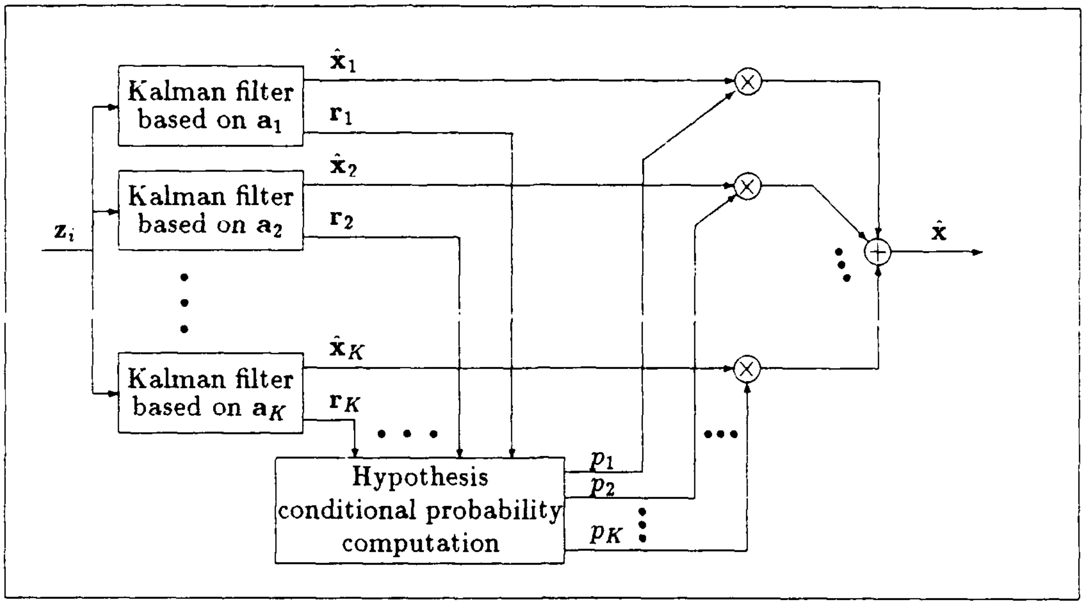

$\mathbf{z}_i$
Kalman filter based on $\mathbf{a}_1$
Kalman filter based on $\mathbf{a}_2$
Kalman filter based on $\mathbf{a}_K$
$\hat{\mathbf{x}}_1$
$\mathbf{r}_1$
$\hat{\mathbf{x}}_2$
$\mathbf{r}_2$
$\hat{\mathbf{x}}_K$
$\mathbf{r}_K$
Hypothesis conditional probability computation
p_1
p_2
p_K
$\hat{\mathbf{x}}$

Figure 2.4. MMAE block diagram [20]

$$f_{\mathbf{z}(t_i)|\mathbf{a},\mathbf{Z}(t_{i-1})}(\mathbf{z}_i|\alpha_k, \mathbf{Z}_{i-1}) = \frac{1}{(2\pi)^{m/2}|\mathbf{A}_k|^{1/2}} exp[-\frac{1}{2}\mathbf{r}_k^T(t_i)\mathbf{A}_k^{-1}(t_i)\mathbf{r}_k(t_i)] \tag{2.18}$$

where:

- $\mathbf{A}_k(t_i) = \mathbf{H}_k(t_i)\mathbf{P}_k(t_i)\mathbf{H}_k^T(t_i) + \mathbf{R}_k(t_i)$
- $\mathbf{r}_k(t_i) = \mathbf{z}_i - \mathbf{H}_k(t_i)\mathbf{x}_k(t_i^-)$
- m = the dimension of the measurement vector.

If an estimate of the parameter vector is required, then a conditional mean of a at the ith sample time produces: [20]

$$\hat{\mathbf{a}}(t_i) \triangleq E[\mathbf{a}(t_i)|\mathbf{Z}(t_i) = \mathbf{Z}_i] = \sum_{k=1}^{K} \mathbf{a}_k p_k(t_i) \tag{2.19}$$

According to Maybeck [20:page 133], the MMAE algorithm performance is dependent on a significant difference between the residuals of the "correct" and "incorrect" filter models. Therefore, during the filter noise tuning process, the dynamics pseudonoise added to account for uncertainties in the model of the system must be kept small. By adding too much pseudonoise, the differences between the good and bad models is masked. [20:page 133]

A procedure used for tuning the filters of a MMAE algorithm was outlined by Lashlee [9]. In Lashlee's work, the tuning process consisted of varying the dynamics noise values **Q** while holding the measurement noise values **R** constant. After the **Q** values were determined , the **R** values were varied while holding the **Q** values constant.

The approach taken by Lashlee was to use a "truth" model and single Kalman filter with the same **Q** and **R** values throughout the tuning process based on the middle of the parameter vector space. The **Q** values were varied one at a time by orders of magnitude until the rms error of the state estimate no longer decreased substantially. The error of the state estimate is the difference between the filter computed state estimate and the true value of the state as produced by the "truth" model. Once the **Q** values were determined, a similar approach was taken to tune the **R** values.

2.7 Multiple Model-Based Control Approach

Tellman combines the MMAE principles with model-based control to develop a new form of adaptive model-based control called Multiple Model-Based Control (MMBC) technique [30]. The MMBC technique assumes that Equation (2.4) can be written in a more general form as [30:Chapter 3]:

$$\ddot{\mathbf{q}}(t) = \mathbf{f}(\mathbf{q}, \dot{\mathbf{q}}, \boldsymbol{\tau}, \mathbf{a}, \mathbf{z}, t) + \mathbf{G}(t)\mathbf{w}(t) \tag{2.20}$$

with an associated measurement, $\mathbf{z}(t)$, model:

$$\mathbf{z}(t) = \mathbf{h}(\mathbf{q}, \dot{\mathbf{q}}, \boldsymbol{\tau}, \mathbf{a}, t) + \mathbf{v}(t) \tag{2.21}$$

where:

- $\mathbf{a}$ = the vector of uncertain parameters (payload in this application).

- $\mathbf{f}()$ and $\mathbf{h}()$ = nonlinear functions of the arguments.
- $\mathbf{w}(t)$ = zero mean, white Gaussian noise to account for uncertainties in the dynamics model.
- $\mathbf{v}(t)$ = zero mean, white Gaussian measurement noise.
- $\mathbf{G}(t)$ = the scaling matrix for the dynamics noise.

The MMBC technique linearized Equations (2.20) and (2.21) about some desired nominal trajectory to develop linearized perturbation equations [30:Chapter 3]. These perturbation equations were used to develop the Kalman filters of a closed-loop form of MMAE called ΔMMAE [15,31].

A block diagram of the ΔMMAE algorithm is shown in Figure 3.3of the next chapter (it will be discussed more fully in that chapter). Similar to the MMAE algorithm, a bank of K Kalman filters is used. However, the ΔMMAE algorithm provides estimates of the variations $\Delta\mathbf{a}$ in the parameter vector, and using a function of the residuals to calculate the sign of $\Delta\mathbf{a}$, a new estimate of the actual parameter vector is calculated by: [15,31]

$$\hat{\mathbf{a}}(t_i) = \hat{\mathbf{a}}(t_{i-1}) + \Delta\mathbf{a}SIGN\{f(\mathbf{r}(t_i))\} \tag{2.22}$$

where $f(\mathbf{r}(t_i)$ is some function of the residuals used in the ΔMMAE used to determine the sign. The actual function may be different for each application.

The initial development of MMBC limited the estimation to the payload mass for a PUMA-560 manipulator based on previous research for model-based control with various payloads conducted by Leahy [11]. A total of three Kalman filters were used. Simulation results showed the potential of the MMBC to converge quickly on an estimate of the mass and the ability to maintain good tracking performance [30:Chapter 4] and [15,31].

A potential problem of the MMBC technique is the need for more Kalman filters as more of the payload parameters need to be estimated (i.e. center of mass) for a manipulator. The increase in Kalman filters results in more computation time and thus algorithm complexity. Although the computational complexity is of some concern, the advances in computer technology can minimize these concerns and allow for experimental evaluations of the algorithm. In order to test the true potential of any control algorithm, an experimental evaluation must be conducted. These experimental evaluations will not only validate simulation results, but the benefits of this research move one step closer to eventual implementation.

2.8 Summary

A short summary of MRAC, adaptive control using an autoregressive model, adaptive perturbation control, and review of the current approaches to adaptive model-based control were presented. These approaches differ in their technique for the adaptation scheme. The approach based on Lyapunov theory guarantees controller stability and the asymptotic convergence of the tracking error to zero. However, the drawback to this approach is the speed of convergence. The second approach, the MMBC technique, is a more recent development based on the principles of Bayesian estimation and use of multiple Kalman filters to perform the estimation of the uncertain parameters. As the number of uncertain parameters increase, the requirement for more Kalman filters results in a more computationally complex algorithm. Advances in computer technology should minimize the computation burden and allow for experimental evaluation to validate simulation results. An evaluation and more detailed development of the original MMBC algorithm based on a closed-loop linearized perturbation dynamics model is presented in the next chapter.

III. Closed-Loop Multiple Model-Based Control Evaluation

3.1 Introduction

An initial objective of this thesis research was to experimentally evaluate the closed-loop multiple model-based control (MMBC) algorithm developed by Tellman [30]. During this evaluation, some errors in the control algorithm software were discovered and subsequently corrected. The result of these corrections required retuning the Kalman filters used in a closed-loop MMAE to estimate payload variation. Exhaustive methods to retune the filters of the closed-loop MMBC algorithm were not successful. Further analysis of the MMBC algorithm provided insight into an alternate design of the Kalman filter equations used in the payload variation estimator. The remainder of this chapter presents the details of Tellman's MMBC algorithm, the software evaluation and corrections, and analysis that led to the alternate design.

From now on the MMBC algorithm developed by Tellman will be referred to as closed-loop MMBC (CL/MMBC). The CL/MMBC nomenclature is being used to distinguish Tellman's algorithm from the alternate design developed in Chapter 4.

3.2 Closed-Loop MMBC Algorithm Development

As discussed in the previous chapter, the nonlinear equations of motion for a n-degrees of freedom rigid robot can be written as a function of payload parameters ($\mathbf{a}$), joint position ($\mathbf{q}$), and joint velocity ($\dot{\mathbf{q}}$):

$$\mathbf{N}\boldsymbol{\tau}(t) = [\mathbf{D}(\mathbf{q},\mathbf{a}) + N^2\mathbf{J}_m]\ddot{\mathbf{q}}(t) + \mathbf{h}(\dot{\mathbf{q}},\mathbf{q},\mathbf{a}) + \mathbf{N}^2\boldsymbol{\beta}_{\mathbf{m}}\dot{\mathbf{q}}(t) + \mathbf{g}(\mathbf{q},\mathbf{a}) + \boldsymbol{\tau}_s \tag{3.1}$$

where the terms are as defined for Equation (2.4). Rearranging Equation (3.1) to get an equality for $\ddot{\mathbf{q}}$ yields the following deterministic nonlinear differential equation:

$$\begin{aligned}\ddot{\mathbf{q}}(t) &= \left[\mathbf{D}(\mathbf{q},\mathbf{a}) + \mathbf{N}^2\mathbf{J}_m\right]^{-1}\left[\mathbf{N}\boldsymbol{\tau}(t) - \mathbf{h}(\dot{\mathbf{q}},\mathbf{q},\mathbf{a}) - \mathbf{N}^2\boldsymbol{\beta}_{\mathbf{m}}\dot{\mathbf{q}} - \mathbf{g}(\mathbf{q},\mathbf{a}) - \boldsymbol{\tau}_s\right] \\ &= \mathbf{f}(\mathbf{q},\dot{\mathbf{q}},\boldsymbol{\tau},\mathbf{a},t) \qquad (3.2)\end{aligned}$$

An exact model of a robotic system is almost impossible to obtain, and Equation (3.2) always has some uncertainties due to modeling inaccuracies or unmodeled disturbances.

One way to account for these uncertainties is to assume they can be modeled as linearly added white Gaussian noise. Equation (3.2) can then be rewritten as a stochastic nonlinear differential equation:

$$\ddot{\mathbf{q}}(t) = \mathbf{f}(\mathbf{q}, \dot{\mathbf{q}}, \boldsymbol{\tau}, \mathbf{a}, t) + \mathbf{G}(t)\mathbf{w}(t) \tag{3.3}$$

where $\mathbf{w}(t)$ is a zero-mean white Gaussian noise process with covariance kernel:

$$E\{\mathbf{w}(t)\mathbf{w}^T(t + \Delta t)\} = \mathbf{Q}(t)\delta(\Delta t) \tag{3.4}$$

Furthermore, if the discrete-time measurements can be modeled in general as a known nonlinear function of $\mathbf{q}$, $\dot{\mathbf{q}}$, $\mathbf{a}$, and plus linearly additive noise corruption, the measurement model would look like:

$$\mathbf{z}(t_i) = \mathbf{h}(\mathbf{q}, \dot{\mathbf{q}}, \mathbf{a}, t_i) + \mathbf{v}(t_i) \tag{3.5}$$

w'ere $\mathbf{v}(t_i)$ is a zero-mean white Gaussian noise sequence with covariance kernel

$$E\{\mathbf{v}(t_i)\mathbf{v}^T(t_j)\} = \mathbf{R}(t_i)\delta_{ij} \tag{3.6}$$

To develop the CL/MMBC algorithm, assume that a desired nominal trajectory satisfies the deterministic differential equation:

$$\ddot{\mathbf{q}}_d(t) = \mathbf{f}(\mathbf{q}_d, \dot{\mathbf{q}}_d, \boldsymbol{\tau}_d, \mathbf{a}, t) \tag{3.7}$$

and associated with the desired nominal trajectory would be the sequence of nominal measurements:

$$\mathbf{z}_d(t_i) = \mathbf{h}(\mathbf{q}_d, \dot{\mathbf{q}}_d, \mathbf{a}, t_i) \tag{3.8}$$

Examining the perturbations from the assumed nominal trajectory yields a stochastic process satisfying:

$$[\ddot{\mathbf{q}}(t) - \ddot{\mathbf{q}}_d(t)] = \mathbf{f}(\mathbf{q}, \dot{\mathbf{q}}, \boldsymbol{\tau}, \mathbf{a}, t) - \mathbf{f}(\mathbf{q}_d, \dot{\mathbf{q}}_d, \boldsymbol{\tau}_d, \mathbf{a}, t) + \mathbf{G}(t)\mathbf{w}(t) \tag{3.9}$$

Expressing Equation (3.9) as a Taylor series expansion about the nominally desired trajectory and nominal torque $\boldsymbol{\tau}_d$, yields:

$$\begin{aligned}[\ddot{\mathbf{q}}(t) - \ddot{\mathbf{q}}_d(t)] \;=\; & \left.\frac{\partial \mathbf{f}(\mathbf{q}, \dot{\mathbf{q}}, \boldsymbol{\tau}, \mathbf{a}, t)}{\partial \mathbf{q}}\right|_d [\mathbf{q} - \mathbf{q}_d] + \left.\frac{\partial \mathbf{f}(\mathbf{q}, \dot{\mathbf{q}}, \boldsymbol{\tau}, \mathbf{a}, t)}{\partial \dot{\mathbf{q}}}\right|_d [\dot{\mathbf{q}} - \dot{\mathbf{q}}_d] \\ & + \left.\frac{\partial \mathbf{f}(\mathbf{q}, \dot{\mathbf{q}}, \boldsymbol{\tau}, \mathbf{a}, t)}{\partial \boldsymbol{\tau}}\right|_d [\boldsymbol{\tau} - \boldsymbol{\tau}_d] + h.o.t. + \mathbf{G}(t)\mathbf{w}(t)\end{aligned} \tag{3.10}$$

A first order approximation to Equation (3.10) results by ignoring the *h.o.t.* terms. If a state vector is defined as:

$$\mathbf{x}(t) = \begin{bmatrix} \mathbf{q}(t) \\ \dot{\mathbf{q}}(t) \end{bmatrix}_{2n\times 1}$$

a first order approximation of Equation (3.10) yields:

$$\boldsymbol{\delta}\dot{\mathbf{x}}_2(t) = \mathbf{F}_2'(\mathbf{x}_d, \boldsymbol{\tau}_d, \mathbf{a}, t)\boldsymbol{\delta}\mathbf{x} + \mathbf{B}_2(\mathbf{x}_d, \boldsymbol{\tau}_d, \mathbf{a}, t)\boldsymbol{\delta\tau} + \mathbf{G}(t)\mathbf{w}(t) \tag{3.11}$$

where:

$$\mathbf{F}_2'(\mathbf{x}_d, \boldsymbol{\tau}_d, \mathbf{a}, t) = \left.\frac{\partial \mathbf{f}(\mathbf{x}, \boldsymbol{\tau}, \mathbf{a}, t)}{\partial \mathbf{x}}\right|_d \tag{3.12}$$

$$\mathbf{B}_2(\mathbf{x}_d, \boldsymbol{\tau}_d, \mathbf{a}, t) = \left.\frac{\partial \mathbf{f}(\mathbf{x}, \boldsymbol{\tau}, \mathbf{a}, t)}{\partial \boldsymbol{\tau}}\right|_d \tag{3.13}$$

$$\boldsymbol{\delta}\dot{\mathbf{x}}_2(t) = [\ddot{\mathbf{q}} - \ddot{\mathbf{q}}_d] \tag{3.14}$$

$$\boldsymbol{\delta\tau} = [\boldsymbol{\tau} - \boldsymbol{\tau}_d] \tag{3.15}$$

$$\boldsymbol{\delta}\mathbf{x} = \begin{bmatrix} \mathbf{q} - \mathbf{q}_d \\ \dot{\mathbf{q}} - \dot{\mathbf{q}}_d \end{bmatrix} \tag{3.16}$$

and $\boldsymbol{\tau}_d$ is the feedforward compensation torques used in the model-based control law (see Equation (2.6)). Since $\boldsymbol{\delta}\mathbf{x}_1(t) = [\mathbf{q} - \mathbf{q}_d]$, the following relationship exists.

$$\boldsymbol{\delta}\dot{\mathbf{x}}_1(t) = \boldsymbol{\delta}\mathbf{x}_2(t) \tag{3.17}$$

Combining Equations (3.11) and (3.17), the linear perturbation state equation for an n degrees of freedom robot manipulator can be written as:

$$\boldsymbol{\delta}\dot{\mathbf{x}}(t) = \mathbf{F}'(\mathbf{x}_d, \boldsymbol{\tau}_d, \mathbf{a}, t)\,\boldsymbol{\delta}\mathbf{x}(t) + \mathbf{B}(\mathbf{x}_d, \boldsymbol{\tau}_d, \mathbf{a}, t)\,\boldsymbol{\delta\tau} + \mathbf{G}(t)\,\mathbf{w}(t) \tag{3.18}$$

where:

$$\mathbf{F}'(\mathbf{x}_d, \boldsymbol{\tau}_d, \mathbf{a}, t) = \left[\begin{array}{c} \mathbf{0}_{n\times n} \mid \mathbf{I}_{n\times n} \\ ------ \\ \left.\frac{\partial \mathbf{f}(\mathbf{x}, \boldsymbol{\tau}, \mathbf{a}, t)}{\partial \mathbf{x}}\right|_d \end{array} \right]_{2n\times 2n} \tag{3.19}$$

$$\mathbf{B}(\mathbf{x}_d, \boldsymbol{\tau}_d, \mathbf{a}, t) = \left[\begin{array}{c} \mathbf{0}_{n\times n} \\ ------ \\ \left.\frac{\partial \mathbf{f}(\mathbf{x}, \boldsymbol{\tau}, \mathbf{a}, t)}{\partial \boldsymbol{\tau}}\right|_d \end{array} \right]_{2n\times n} \tag{3.20}$$

A block diagram of Equation (3.18) is illustrated in Figure (3.1).

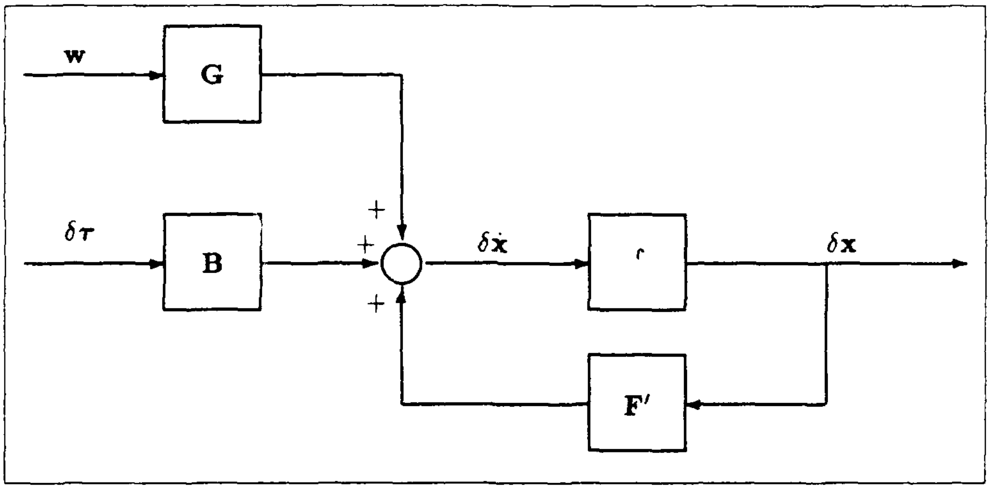


Figure 3.1. Block Diagram of Linear Perturbation Model

Assuming $\delta\boldsymbol{\tau}$ can be written as the PD feedback torques of the model-based control law:

$$\begin{aligned} \delta\boldsymbol{\tau} &= -\left[\mathbf{K}_{\mathbf{p}_{n\times n}} \mid \mathbf{K}_{\mathbf{v}_{n\times n}}\right] \delta\mathbf{x} \\ &= -\mathbf{K}\,\delta\mathbf{x} \end{aligned} \tag{3.21}$$

where:

- $\mathbf{K_p} = n \times n$ diagonal constant position gains matrix.
- $\mathbf{K_v} = n \times n$ diagonal constant velocity gains matrix.

then Equation (3.18) can be written as:

$$\begin{aligned}\delta\dot{\mathbf{x}}(t) &= [\mathbf{F}'(\mathbf{x}_d, \boldsymbol{\tau}_d, \mathbf{a}, t) - \mathbf{B}(\mathbf{x}_d, \boldsymbol{\tau}_d, \mathbf{a}, t)\,\mathbf{K}]\delta\mathbf{x}(t) + \mathbf{G}(t)\,\mathbf{w}(t) \\ &= \mathbf{F}(\mathbf{x}_d, \boldsymbol{\tau}_d, t)\,\delta\mathbf{x}(t) + \mathbf{G}(t)\,\mathbf{w}(t) \qquad (3.22)\end{aligned}$$

and the discrete measurement model associated with Equation (3.22) is:

$$\delta\mathbf{z}(t_i) = \mathbf{H}\,\delta\mathbf{x}(t_i) + \mathbf{v}(t_i) \qquad (3.23)$$

where

- $\mathbf{H}$ = the $m \times 2n$ measurement transformation matrix.
- $\mathbf{v}(t_i)$ = m-dimensional zero-mean white Gaussian measurement noise vector.

Tellman assumed that the white Gaussian system dynamics noise entered the system at the same point as the perturbation torques $\delta\boldsymbol{\tau}$ and this resulted in $\mathbf{G}(t) = \mathbf{B}(\mathbf{x}_d, \boldsymbol{\tau}_d, \mathbf{a}, t)$ [30:page 4–3]. Therefore, Figure 3.1 can be rearranged as shown in Figure 3.2.

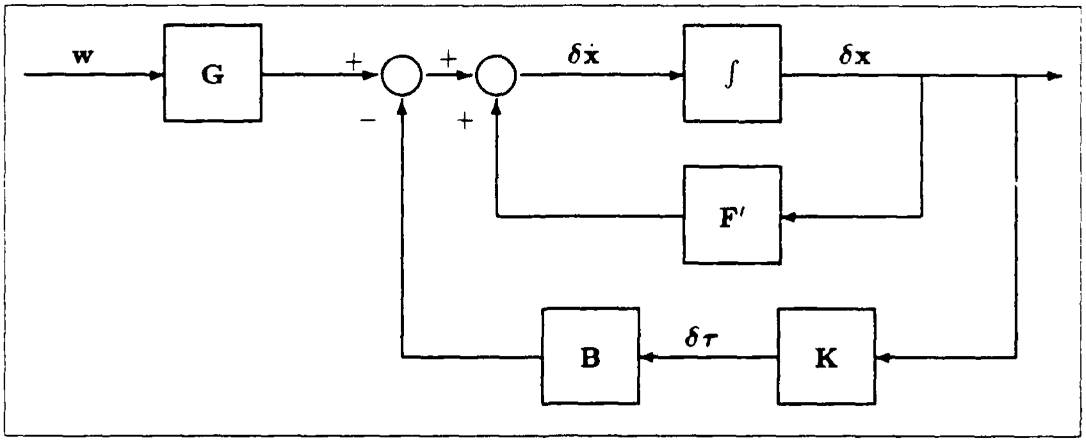


Figure 3.2. Rearranged Block Diagram of Linear Perturbation Model

Tellman used Equations (3.22) and (3.23) to develop the Kalman filters of a closed-loop form of the multiple model adaptive estimation (MMAE) algorithm. The closed-loop MMAE algorithm is called ΔMMAE [15,31] and consists of a bank of Kalman filters each based on a particular value of the uncertain parameter vector ($\mathbf{a}_1$, $\mathbf{a}_2$, ... , $\mathbf{a}_K$). The Kalman filter equations based on $\mathbf{a}_k$ can be written as:

$$\delta\hat{\mathbf{x}}_k(t_{i+1}^-) = \mathbf{\Phi}_k(t_{i+1}, t_i)\delta\hat{\mathbf{x}}_k(t_i^+) \tag{3.24}$$

$$\mathbf{P}_k(t_{i+1}^-) = \mathbf{\Phi}_k(t_{i+1}, t_i)\mathbf{P}_k(t_i^+)\mathbf{\Phi}_k^T(t_{i+1}, t_i) + \mathbf{Q}_{d_k}(t_i) \tag{3.25}$$

$$\delta\hat{\mathbf{x}}_k(t_i^+) = \delta\hat{\mathbf{x}}_k(t_i^-) + \mathbf{K}_k(t_i)\left[\delta\mathbf{z}_i - \mathbf{H}_k(t_i)\delta\hat{\mathbf{x}}_k(t_i^-)\right] \tag{3.26}$$

$$\mathbf{P}_k(t_i^+) = \mathbf{P}_k(t_i^-) - \mathbf{K}_k(t_i)\mathbf{H}_k(t_i)\mathbf{P}_k(t_i^-) \tag{3.27}$$

$$\mathbf{K}_k(t_i) = \mathbf{P}_k(t_i^-)\mathbf{H}_k^T(t_i)\left[\mathbf{H}_k(t_i)\mathbf{P}_k(t_i^-)\mathbf{H}_k^T(t_i) + \mathbf{R}_k(t_i)\right]^{-1} \tag{3.28}$$

where:

- $\delta\hat{\mathbf{x}}_k(t_i^-)$ = the estimate of the perturbation state vector based on a_k at time t_i just prior to the measurement update at time t_i.
- $\delta\hat{\mathbf{x}}_k(t_i^+)$ = the updated estimate of the perturbation state vector based on a_k at time t_i just after the measurement update at time t_i.
- $\mathbf{P}_k(t_i^-)$ = the covariance matrix based on a_k prior to the measurement update at time t_i.
- $\mathbf{P}_k(t_i^+)$ = the covariance matrix based on a_k after the measurement update at time t_i.
- $\delta\mathbf{z}(t_i)$ = the vector of noise corrupted measurements of the perturbation states at time t_i.
- $\mathbf{H}_k(t_i)$ = the measurement matrix of the kth filter at time t_i.
- $\mathbf{K}_k(t_i)$ = the Kalman filter gain matrix based on a_k at time t_i.
- $\mathbf{\Phi}_k(t_{i+1}, t_i)$ = the state transition matrix based on $\mathbf{a}_k$ that satisfies $\dot{\mathbf{\Phi}}_k(t_{i+1}, t_i) = \mathbf{F}(\mathbf{x}_d, \mathbf{r}_d, a_k, t)\mathbf{\Phi}_k(t_{i+1}, t_i)$.
- $\mathbf{R}_k(t_i)$ = the matrix of measurement noise covariance for the kth filter.
- $\mathbf{Q}_{d_k}(t_i)$ = the matrix of dynamics noise covariance for the kth filter.

A block diagram of the ΔMMAE algorithm is shown in Figure 3.3. As Figure 3.3 illustrates, the measurement $\boldsymbol{\delta}\mathbf{z}_i$, at the ith sample time, is passed to each of the Kalman filters. The residuals ($\mathbf{r}_1$, $\mathbf{r}_2$, ... , $\mathbf{r}_K$) generated by the K filters are passed to the hypothesis conditional probability. These hypothesis conditional probabilities are used as weighting factors to generate estimates of the parameter variations, and the hypothesis conditional probabilities can be computed as: [20]

$$p_k(t_i) = \frac{f_{\boldsymbol{\delta}\mathbf{z}(t_i)|\mathbf{a},\boldsymbol{\delta}\mathbf{Z}(t_{i-1})}(\boldsymbol{\delta}\mathbf{z}_i|\boldsymbol{\alpha}_k, \boldsymbol{\delta}\mathbf{Z}_{i-1})\, p_k(t_{i-1})}{\sum_{j=1}^{K} f_{\boldsymbol{\delta}\mathbf{z}(t_i)|\mathbf{a},\boldsymbol{\delta}\mathbf{Z}(t_{i-1})}(\boldsymbol{\delta}\mathbf{z}_i|\boldsymbol{\alpha}_j, \boldsymbol{\delta}\mathbf{Z}_{i-1})\, p_j(t_{i-1})} \tag{3.29}$$

where:

- $p_k(t_i)$ = the hypothesis conditional probability associated with $\mathbf{a}_k$ at time t_i.
- $\boldsymbol{\delta}\mathbf{Z}(t_{i-1})$ = the measurement history up through time t_{i-1}.

To compute the hypothesis conditional probabilities of Equation (3.29) requires the evaluation of $f_{\boldsymbol{\delta}\mathbf{z}(t_i)|\mathbf{a},\boldsymbol{\delta}\mathbf{Z}(t_{i-1})}(\boldsymbol{\delta}\mathbf{z}_i|\boldsymbol{\alpha}_k, \boldsymbol{\delta}\mathbf{Z}_{i-1})\, p_k(t_{i-1})$ and can be evaluated as: [20]

$$f_{\boldsymbol{\delta}\mathbf{z}(t_i)|\mathbf{a},\boldsymbol{\delta}\mathbf{Z}(t_{i-1})}(\boldsymbol{\delta}\mathbf{z}_i|\boldsymbol{\alpha}_k, \boldsymbol{\delta}\mathbf{Z}_{i-1}) = \frac{1}{(2\pi)^{m/2}|\mathbf{A}_k|^{1/2}} exp\left[-\frac{1}{2}\mathbf{r}_k^T(t_i)\,\mathbf{A}_k^{-1}(t_i)\,\mathbf{r}(t_i)\right] \tag{3.30}$$

where:

- $\mathbf{A}_k(t_i) = \mathbf{H}_k(t_i)\,\mathbf{P}(t_i^-)\,\mathbf{H}_k^T(t_i) + \mathbf{R}_k(t_i)$
- $\mathbf{r}_k(t_i) = \boldsymbol{\delta}\mathbf{z}_i - \mathbf{H}_k(t_i)\,\hat{\boldsymbol{\delta}\mathbf{x}}(t_i^-)$
- m = the dimension of the measurement vector.

Using the hypothesis conditional probabilities calculated by Equation (3.29), the estimate of the parameter variations can be calculated as:[30]

$$\boldsymbol{\Delta}\hat{\mathbf{a}}(t_i) = \sum_{k=1}^{K} \mathbf{a}_k p_k(t_i) \tag{3.31}$$

where $\boldsymbol{\Delta}\hat{\mathbf{a}}(t_i)$ is the parameter variation estimate at time t_i.

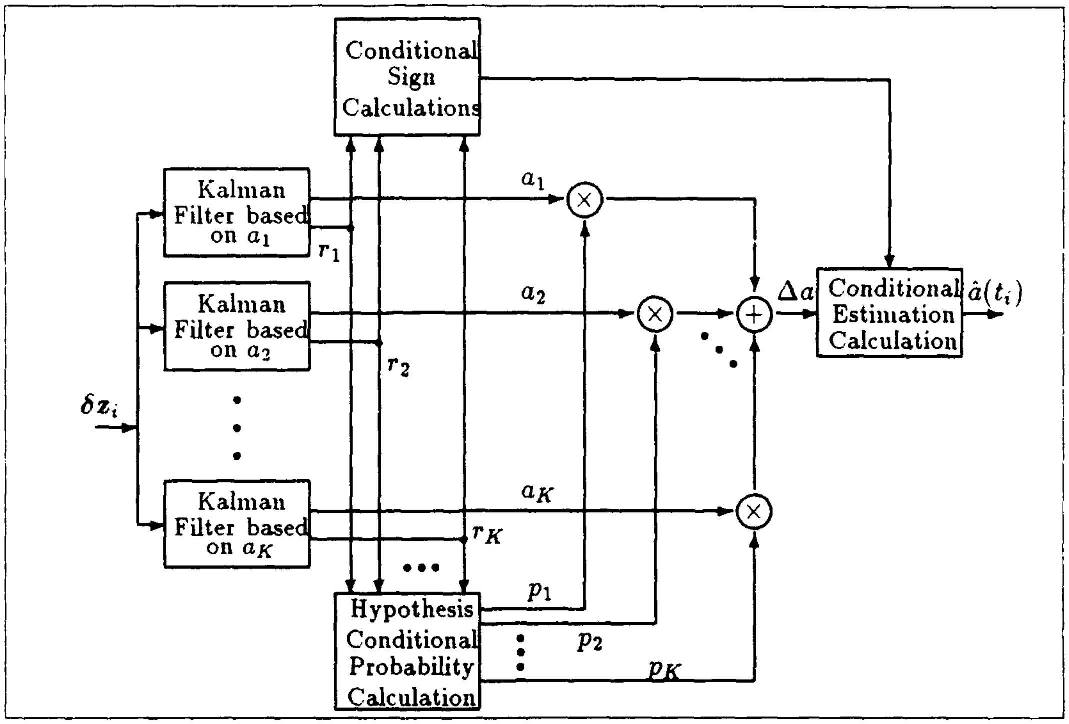


Figure 3.3. ΔMMAE Block Diagram [30]

An estimate of the actual parameters is computed by using the sign of the residuals to determine whether to add or subtract the $\Delta\hat{\mathbf{a}}$ to the previous parameter estimate. This calculation can be written as: [30]

$$\hat{\mathbf{a}}(t_i) = \hat{\mathbf{a}}(t_{i-1}) + \Delta\hat{\mathbf{a}}(t_i)\, SIGN\{f(\mathbf{r}(t_i))\} \tag{3.32}$$

where $f(\mathbf{r}(t_i))$ is a function of the residuals and may be different for each application. For example, Tellman's function of the residuals was just the joint 2 residual associated with the filter based on an assumed payload of 2.5 Kg [30].

The structure of the CL/MMBC algorithm combines the ΔMMAE algorithm with a model-based controller as shown in Figure 3.4. Tellman evaluated the CL/MMBC algorithm by using the first three links of a PUMA-560 manipulator as a test case. The uncertain parameter to be estimated was limited to payload mass, and three discrete values were used (a_1 = 0.0 Kg, a_2 = 2.5 Kg, and a_3 = 5.0 Kg).

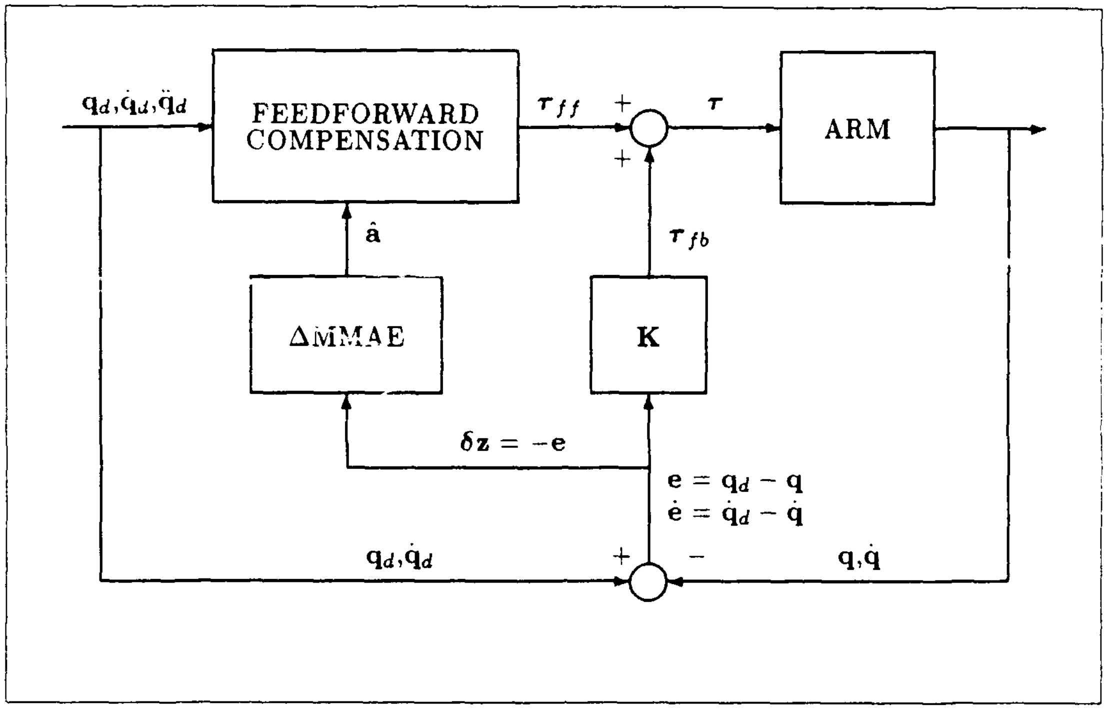

$\mathbf{q}_d, \dot{\mathbf{q}}_d, \ddot{\mathbf{q}}_d$
FEEDFORWARD COMPENSATION
τ_{ff}
+
τ
ARM
+
$\hat{\mathbf{a}}$
τ_{fb}
ΔMMAE
K
$\delta\mathbf{z} = -\mathbf{e}$
$\mathbf{e} = \mathbf{q}_d - \mathbf{q}$
$\dot{\mathbf{e}} = \dot{\mathbf{q}}_d - \dot{\mathbf{q}}$
$\mathbf{q}_d, \dot{\mathbf{q}}_d$
+
−
$\mathbf{q}, \dot{\mathbf{q}}$

Figure 3.4. Multiple Model-Based Control Algorithm Block Diagram

For Tellman's case study the $\mathbf{\Delta a}$ estimate was biased and required the following calculation based on a least-squares curve fit of the biased parameter variation estimate data [30:4-19]:

$$\Delta\hat{a}_f(t_i) = 1.856\Delta\hat{a}(t_i) - 3.793 \tag{3.33}$$

where:

- $\Delta\hat{a}_f(t_i)$ = the curve fit estimate of the parameter variation.
- $\Delta\hat{a}(t_i)$ = the biased parameter variation estimate.

Tellman implemented Equations (3.32) and (3.33) within the conditional estimation calculation block of Figure 3.3. To implement the algorithm for the PUMA-560 case study required the development of numerous software programs. An evaluation of that software was the starting point of this research effort.

3.3 Software Evaluation

During the process of evaluating the CL/MMBC software, errors were discovered in the following areas:

1. calculation of the robot manipulator dynamics,
2. calculation of $\mathbf{F}(\mathbf{x}_d, \boldsymbol{\tau}_d, t)$ of Equation (3.22), and
3. calculation of $\mathbf{P}(t_i^+)$ of Equation (3.27).

3.3.1 Robot Dynamics Equations The errors in the dynamics calculation of the robot manipulator were a result of undefined variables and incorrect calculation of the Coriolis and centripetal terms of Equation (3.1). The undefined variables were a result of switching the letters of the desired joint velocity variable name (DQ versus QD). Since the variables were never defined, the content was set to zero. This resulted in setting some values of the dynamics to zero. The manipulator dynamics software is based on the PUMA dynamic equations proposed by Tarn [29]. The incorrectly calculated Coriolis and centripetal terms $\mathbf{h}(\mathbf{x}_d, \boldsymbol{\tau}_d, \mathbf{a}, t)$ were discovered by comparing the dynamics equations used in the software with the original equations contained in the Tarn report. These errors resulted in an incorrect model for the PUMA-560 manipulator, and recent analysis shows the importance of accurate Coriolis and centripetal modeling [12].

3.3.2 Plant Matrix Calculation The plant matrix of the linearized perturbation model shown in Equation (3.22) required calculation of partial derivatives of the nonlinear dynamics equation shown in Equation (3.2) with respect to the joint positions and velocities. In order to calculate these partial derivatives, a commercial software package (MACSYMA) that manipulates symbolic equations and develops the software code was used [28]. Tellman developed a batch routine that uses MACSYMA commands to perform the partial derivative equations required in Equation (3.22). Within the MACSYMA batch routine, the dynamics equations for the first three links of the PUMA-560 robot manipulator were required.

When comparing the dynamics equations used in the MACSYMA batch routine with those developed by Tarn, some discrepancies were discovered in the MACSYMA batch routine. These discrepancies were corrected. The significance of these errors may have resulted in a plant matrix with little dependence on the payload parameter and made tuning the Kalman filters difficult. The tuning process is discussed later on in this section.

3.3.3 Update Covariance Calculations The software routine that calculated $\mathbf{P}(t_i^+)$ of Equation (3.27) was in error. The first row, elements $P_{12}(t_i^+)$ through $P_{16}(t_i^+)$, of $\mathbf{P}(t_i^+)$ were zero while the corresponding elements of the first column were not. This is not correct since the covariance matrix is a symmetric matrix. Corrections were made to the software. The errors in the calculation of $\mathbf{P}(t_i^+)$ could have an effect on the performance of of the Kalman filters which would effect the estimation of the payload.

3.3.4 Filter Tuning After correcting the software errors, the payload parameter estimates degraded from those obtained during Tellman's research. Since the corrections were a result of dynamics errors and resulted in a different dynamics model than that used in Tellman's research, it was assumed that the noise levels of the filters needed to be retuned.

The initial procedure used to tune the filters was the same as that used by Tellman. To tune the filters, the PUMA-560 was simulated carrying an external payload of 4.0 Kg while attempting to regulate around a stationary position for 1.5 seconds. The stationary position used for the three joints were -50, -135, and 135 degrees respectively. During the simulation run, the feedforward compensation block was not informed of the payload estimate provided by the ΔMMAE block and assumed that the arm was not carrying any payload. Monitoring the payload estimate provided by the estimator, the filters were tuned to provide the best estimate of the payload variation of 4.0 Kg. The payload variation is defined as the difference between the actual payload and the payload information provided to the feedforward compensation block. Therefore for the tuning process, this variation would be 4.0 Kg since the feedforward assumed the manipulator was not carrying any payload and the actual payload simulated was 4.0 Kg.

After Tellman obtained the best estimate of the 4.0 Kg payload variation, external payloads of 0.5, 1.5, 2.5, 3.5, and 4.5 Kg were simulated and the feedforward compensation block still assumed no payload was on the manipulator. The payload variation estimates provided by the ΔMMAE were recorded. Baed on these estimates, a least-squares curve fit of the data was performed to account for any biased estimates.

A major portion of this research effort was spent trying to retune the filters without success. Following the procedures described above, a good estimate of the 4.0 Kg parameter variation estimate was achieved. However, when other payloads were simulated, the parameter variation estimate did not change. The tuning process described above tried to regulate the manipulator to stay at a fixed position. Since this type of trajectory provides very little excitement of the nonlinear dynamic effects inherent of a manipulator, other trajectory profiles were tried which moved the manipulator.

The first trajectory denoted as Trajectory One used a minimum-jerk trajectory profile that moved the first three joints of the manipulator from a starting point of (−50°, −135°, 135°) to (50°, −85°, 30°) in 2.0 seconds. The other trajectory profile used a minimum-jerk trajectory profile and was denoted as Trajectory Two. Trajectory Two moved the first three joints of the manipulator through the same motion as Trajectory One but in 1.5 seconds. Both of these trajectory profiles are contained in Appendix A.

Using these two trajectories and the tuning procedures described above, attempts were made to retune the filters and proved unsuccessful. A final attempt was made to retune the filters by using experimental tracking error data. The experimental tracking error data were obtained using a model-based controller not informed of the payload. The ΔMMAE software was modified to use the experimental data instead of the simulator data. Again the retuning was unsuccessful. The inability to retune the filters successfully, prompted further analysis of the MMBC algorithm in an attempt to gain insights into a modification to allow for better tuning.

3.4 Analysis

The linear perturbation model developed in the Section 3.2 was based on the model-based control structure. As discussed in Chapter 2, the model-based controller is divided into a feedforward and feedback control law. The feedforward partition calculates the nominal torque required to follow some desired trajectory based on a nonlinear model of the manipulator (see Equation (2.4)) which is a function of payload. Since Equation (2.4) is only a model, uncertainties exist due to modeling inaccuracies and unmodeled disturbances. Therefore, a PD controller is used in the feedback partition to calculate the feedback torques to minimize errors from the desired trajectory. The PD controller was used based on the tracking performance obtained in previous research of model-based controllers [10,11,14]. The overall torque applied to the manipulator is the sum of the feedforward and feedback torques.

The development of the linear perturbation equations was based on perturbations about some nominally desired trajectory and torque. The nominally desired torque used in the linear perturbation model was assumed to be the feedforward torque, based on knowledge of the payload parameter, as computed in Equation (2.6). Therefore, the plant matrix ($\mathbf{F}(\mathbf{x}_d, \boldsymbol{\tau}_d, \mathbf{a}, t)$) calculations for each of the filters was based on a different nominal torque ($\boldsymbol{\tau}_d$). The plant matrix of the linear perturbation model also incorporated the perturbation torques of Equation (3.15) by assuming they could be written as shown in Equation (3.21).

As part of Tellman's research, an analysis of the eigenvalues of the **F** matrix based on payload values of 0.0 and 5.0 Kg was conducted. This analysis revealed that the linearized model was a function of the desired trajectory and had a weak dependence on the payload parameter [30:4–5]. The weak dependence was a result of including the PD feedback loop of the model-based controller as part of the linearized plant matrix. Only when the external payload differed from that used by the controller were differences between each filter's model apparent [30:4–18].

As Figure 3.4 shows, the CL/MMBC feedforward torques are calculated based on the current payload parameter estimate provided by the ΔMMAE block. Therefore, the feedforward torque will change based on the payload estimate provided at the current time and will only provide a "correct" nominal torque when the payload estimate is equal to the external payload on the arm. Remember that each filter model was based on a different nominal torque that assumed the payload was known for the entire trajectory. Therefore, the filters based on payloads not equal to the actual payload will have nominal torques that differ from the feedforward torque when an accurate estimate of the payload is provided. These differences were not accounted for in the closed-loop algorithm proposed by Tellman.

Discussions with Maybeck gave insights into a way to account for the differences between the nominal torques used by the filters and the feedforward torque calculated based on the current estimate of the payload [21]. Maybeck suggested providing a correction term to the filters. This correction term was the difference between the actual feedforward torque being applied and the nominally assumed torque of the filter model. Before attempting to provide these correction terms to the filters, simulations of a PUMA-560 manipulator were conducted using a single model-based control (SMBC) algorithm that could be artificially informed of the payload in order to investigate the effect of assuming the perturbation torques were the PD feedback loop.

Using the two trajectory profiles (Trajectory One and Two) described earlier in Section 3.3.4, an artificially informed SMBC was simulated for external payloads of 0.0, 2.5, and 5.0 Kg. Additionally, simulations of an uninformed SMBC for the 2.5 and 5.0 Kg payloads were performed. Plots of the feedforward torques, commanded torques, and position errors are contained in Appendix A.

The torque profile plots show that the commanded torques (with and without payload information) were similar to the feedforward torque with payload knowledge. Therefore, even without payload information, the commanded torque applied to the manipulator attempted to match the feedforward torque informed of the payload. This indicates that the perturbations between the commanded torque and the nominally desired torque of the linearized model would be smallest for the filter model whose assumed payload closely matches the actual unknown payload. For example if the actual payload was 3.0 Kg, then the filter model based on $a = 2.5$ Kg would have the smallest perturbation torques. Therefore, it might be more appropriate not to assume the perturbation torques are the PD feedback torques.

Additionally, the measured position error profiles for the cases when the controller had knowledge of the payload are shown in Figures A.5 and A.11 for the two respective trajectories. These figures show very little difference between the error profiles when the controller is artificially informed of the payload. However, the position error profiles did differ significantly for the cases when the controller was not informed of the payload information and plots of these errors are shown in Figures A.6 and A.12 for the two respective trajectories. These profiles show that the tracking performance degrades when the controller does not know the payload.

For the corrected dynamics of the PUMA-560, the Kalman filter models used in the ΔMMAE algorithm proposed by Tellman had only a slight dependence on the payload. The sight dependence on the payload equates to very little difference between the filter models. If the filter models do not differ significantly, then the MMAE algorithm will not be able to correctly determine the payload estimate. This indicates that the CL/MMBC has limited applications and is inappropriate for manipulators like the PUMA-560 whose dynamics do not depend heavily on the payload but whose tracking performance does.

3.5 Summary

Within this chapter the details of the originally proposed MMBC algorithm, the corrections to errors discovered during the software evaluation, and the analysis that led to an alternate design were presented. The corrections to the software resulted in a different dynamics model than that used during Tellman's research. Attempts to retune the filter noises were unsuccessful and required a further analysis of the original MMBC algorithm.

For the corrected dynamics model of the PUMA-560 manipulator, the payload could not be distinguished by the filters of the ΔMMAE and made it difficult to estimate the payload. This slight dependence was a result of using the PD feedback loop torques as part of the linearized plant matrix. To use MMAE based techniques for adaptive model-based robot control of a PUMA-560, an alternate solution was required. Based on the discussions with Maybeck and the results obtained from comparisons of the commanded and feedforward torques, a new form of the MMBC algorithm was developed and is called open-loop MMBC. The development is presented in the next chapter.

IV. Open-Loop Multiple Model-Based Control Algorithm Development

4.1 Introduction

The original form of Multiple Model-Based Control (MMBC) proposed by Tellman is limited to manipulators whose dynamics and tracking depend heavily on the payload. The PUMA-560 manipulator is not one of those manipulators. Therefore, a new form of MMBC was developed based on a modification to the linearized perturbation models used for payload estimation. This modification eliminated the use of PD feedback torques as the driving perturbation torques of the linearized perturbation model. The perturbation torque of each filter was assumed to be the difference between the actual commanded torque applied to the arm and the nominal torque assumed by the filter model. This modification will produce sufficient differences in the filters used in a MMAE algorithm and allow for estimation of the payload. The details of the open-loop MMBC development follow.

4.2 Linearized Perturbation Model

Following the same development of Equations (3.1) through (3.18), the linearized perturbation state equation model can be written as

$$\delta\dot{\mathbf{x}}(t) = \mathbf{F}'(\mathbf{x}_d, \boldsymbol{\tau}_d, \mathbf{a}, t)\,\delta\mathbf{x}(t) + \mathbf{B}(\mathbf{x}_d, \boldsymbol{\tau}_d, \mathbf{a}, t)\,\delta\boldsymbol{\tau} + \mathbf{G}(t)\,\mathbf{w}(t) \tag{4.1}$$

with associated discrete measurement model

$$\delta\mathbf{z}(t) = \mathbf{H}(t_i)\delta\mathbf{x}(t_i) + \mathbf{v}(t_i) \tag{4.2}$$

and where:

$$\mathbf{x}(t) = \begin{bmatrix} \mathbf{q}(t) \\ \dot{\mathbf{q}}(t) \end{bmatrix}_{2n\times 1} \tag{4.3}$$

$$\mathbf{F}'(\mathbf{x}_d, \boldsymbol{\tau}_d, \mathbf{a}, t) = \begin{bmatrix} \mathbf{0}_{n\times n} \mid \mathbf{I}_{n\times n} \\ - - - - - \\ \left.\frac{\partial \mathbf{f}(\mathbf{x}, \boldsymbol{\tau}, \mathbf{a}, t)}{\partial \mathbf{x}}\right|_d \end{bmatrix}_{2n\times 2n} \tag{4.4}$$

$$\mathbf{B}(\mathbf{x}_d, \boldsymbol{\tau}_d, \mathbf{a}, t) = \begin{bmatrix} 0_{n\times n} \\ ----- \\ \left.\frac{\partial \mathbf{f}(\mathbf{x}, \boldsymbol{\tau}, \mathbf{a}, t)}{\partial \boldsymbol{\tau}}\right|_d \end{bmatrix}_{2n\times n} \tag{4.5}$$

$$\boldsymbol{\delta}\mathbf{x} = \begin{bmatrix} \mathbf{q} - \mathbf{q}_d \\ \dot{\mathbf{q}} - \dot{\mathbf{q}}_d \end{bmatrix} \tag{4.6}$$

$$\boldsymbol{\delta\tau} = [\boldsymbol{\tau} - \boldsymbol{\tau}_d] \tag{4.7}$$

$$\mathbf{G}(t) = \mathbf{B}(\mathbf{x}_d, \boldsymbol{\tau}_d, \mathbf{a}, t) \tag{4.8}$$

The subtle difference between Equation (4.1) and the linear perturbation state model used by Tellman (see Equation (3.22)) is that the PD feedback loop of the model-based controller is no longer assumed equal to the perturbation torque $\boldsymbol{\delta\tau}$. Therefore, the filters whose assumed payload differ significantly from the actual payload on the manipulator will have larger perturbation torques than the filter whose assumed payload is closest. These differences in the perturbation torques make it possible to estimate the payload. The linearized perturbation model is now in an open-loop formulation which explains the name open-loop MMBC.

4.3 Kalman Filter Equations

Equations (4.1) and (4.2) are in a form for developing linear Kalman filters. Assuming that the uncertain parameters **a** can be discretized into K possible values over the continuous parameter space, a bank of K Kalman filters can be developed to form a multiple model adaptive estimator (MMAE). The MMAE algorithm was briefly reviewed in Chapter 2 and a more detailed discussion can be found in [20:129–136]. Kalman filter equations based on $\mathbf{a}_k$ can be written as:

$$\boldsymbol{\delta}\hat{\mathbf{x}}_k(t_{i+1}^-) = \boldsymbol{\Phi}_k(t_{i+1}, t_i)\,\boldsymbol{\delta}\hat{\mathbf{x}}_k(t_i^+) + \int_{t_i}^{t_{i+1}} \boldsymbol{\Phi}_k(t_{i+1}, \xi)\,\mathbf{B}_k(\mathbf{x}_d, \boldsymbol{\tau}_d, \mathbf{a}_k, \xi)\,\boldsymbol{\delta\tau}_k(\xi)d\xi \tag{4.9}$$

$$\mathbf{P}_k(t_{i+1}^-) = \boldsymbol{\Phi}_k(t_{i+1}, t_i)\,\mathbf{P}_k(t_i^+)\,\boldsymbol{\Phi}_k^T(t_{i+1}, t_i) + \mathbf{Q}_{d_k}(t_i) \tag{4.10}$$

$$\boldsymbol{\delta}\hat{\mathbf{x}}_k(t_i^+) = \boldsymbol{\delta}\hat{\mathbf{x}}_k(t_i^-) + \mathbf{K}_k(t_i)\left[\boldsymbol{\delta}\mathbf{z}_i - \mathbf{H}_k(t_i)\,\boldsymbol{\delta}\hat{\mathbf{x}}_k(t_i^-)\right] \tag{4.11}$$

$$\mathbf{P}_k(t_i^+) = \mathbf{P}_k(t_i^-) - \mathbf{K}_k(t_i)\mathbf{H}_k(t_i)\mathbf{P}_k(t_i^-) \tag{4.12}$$

$$\mathbf{K}_k(t_i) = \mathbf{P}_k(t_i^-)\mathbf{H}_k^T(t_i)\Big[\mathbf{H}_k(t_i)\mathbf{P}_k(t_i^-)\mathbf{H}_k^T(t_i) + \mathbf{R}_k(t_i)\Big]^{-1} \tag{4.13}$$

where the terms are defined in the previous chapter for Equations (3.24) through (3.28) and:

$$\delta\boldsymbol{\tau}_k(t_i) = [\boldsymbol{\tau}(t_i) - \boldsymbol{\tau}_{d_k}(t_i)] \tag{4.14}$$

$$\boldsymbol{\tau}_{d_k}(t_i) = [\hat{\mathbf{D}}(\mathbf{q},\mathbf{a}_k) + \mathbf{N}^2\mathbf{J}_m]\ddot{\mathbf{q}} + \hat{\mathbf{h}}(\dot{\mathbf{q}},\mathbf{q},\mathbf{a}_k) + \mathbf{N}^2\mathbf{B}_m\dot{\mathbf{q}} + \boldsymbol{\tau}_s + \hat{\mathbf{g}}(\mathbf{q},\mathbf{a}_k) \tag{4.15}$$

The differences between these Kalman filter equations and those in the previous chapter are the addition of the integral term in Equation (4.9) and the state transition matrix $\boldsymbol{\Phi}$. The $\boldsymbol{\Phi}$ used in these filter equations is based on the open-loop linearized plant matrix $\mathbf{F}'$ of Equation (4.4) versus the closed-loop linearized plant matrix $\mathbf{F}$ of Equation (3.22). Also, note that $\boldsymbol{\tau}_{d_k}$ is the same as the feedforward torque $\boldsymbol{\tau}_{ff}$ based on $\mathbf{a}_k$ for a model-based controller.

4.4 Parameter Estimation

Using a MMAE algorithm based on these Kalman filters, a Bayesian estimate of the uncertain payload parameter can be obtained by computing

$$\hat{\mathbf{a}}(t_i) = \sum_{k=1}^{K} \mathbf{a}_k p_k(t_i) \tag{4.16}$$

where:

$$p_k(t_i) = \frac{f_{\boldsymbol{\delta}\mathbf{z}(t_i)|\mathbf{a},\boldsymbol{\delta}\mathbf{Z}(t_{i-1})}(\boldsymbol{\delta}\mathbf{z}_i|\boldsymbol{\alpha}_k, \boldsymbol{\delta}\mathbf{Z}_{i-1})p_k(t_{i-1})}{\sum_{j=1}^{K} f_{\boldsymbol{\delta}\mathbf{z}(t_i)|\mathbf{a},\boldsymbol{\delta}\mathbf{Z}(t_{i-1})}(\boldsymbol{\delta}\mathbf{z}_i|\boldsymbol{\alpha}_j, \boldsymbol{\delta}\mathbf{Z}_{i-1})p_j(t_{i-1})} \tag{4.17}$$

Note that the MMAE estimates payload and not payload variations as the original MMBC algorithm did. To compute $p_k(t_i)$ of Equation (4.17) requires the evaluation of the conditional density function $f_{\boldsymbol{\delta}\mathbf{z}(t_i)|\mathbf{a},\boldsymbol{\delta}\mathbf{Z}(t_{i-1})}(\boldsymbol{\delta}\mathbf{z}_i|\boldsymbol{\alpha}_k, \boldsymbol{\delta}\mathbf{Z}_{i-1})$. This can be evaluated as: [20]

$$f_{\boldsymbol{\delta}\mathbf{z}(t_i)|\mathbf{a},\boldsymbol{\delta}\mathbf{Z}(t_{i-1})}(\boldsymbol{\delta}\mathbf{z}_i|\alpha_k, \boldsymbol{\delta}\mathbf{Z}_{i-1}) = \frac{1}{(2\pi)^{m/2}|\mathbf{A}_k|^{1/2}} exp[-\frac{1}{2}\mathbf{r}_k^T(t_i)\mathbf{A}_k^{-1}(t_i)\mathbf{r}_k(t_i)] \tag{4.18}$$

where:

- $\mathbf{A}_k(t_i) = \mathbf{H}_k(t_i)\mathbf{P}_k(t_i^-)\mathbf{H}_k^T(t_i) + \mathbf{R}_k(t_i)$
- $\mathbf{r}_k(t_i) = \boldsymbol{\delta}\mathbf{z}_i - \mathbf{H}_k(t_i)\boldsymbol{\delta}\hat{\mathbf{x}}_k(t_i^-)$

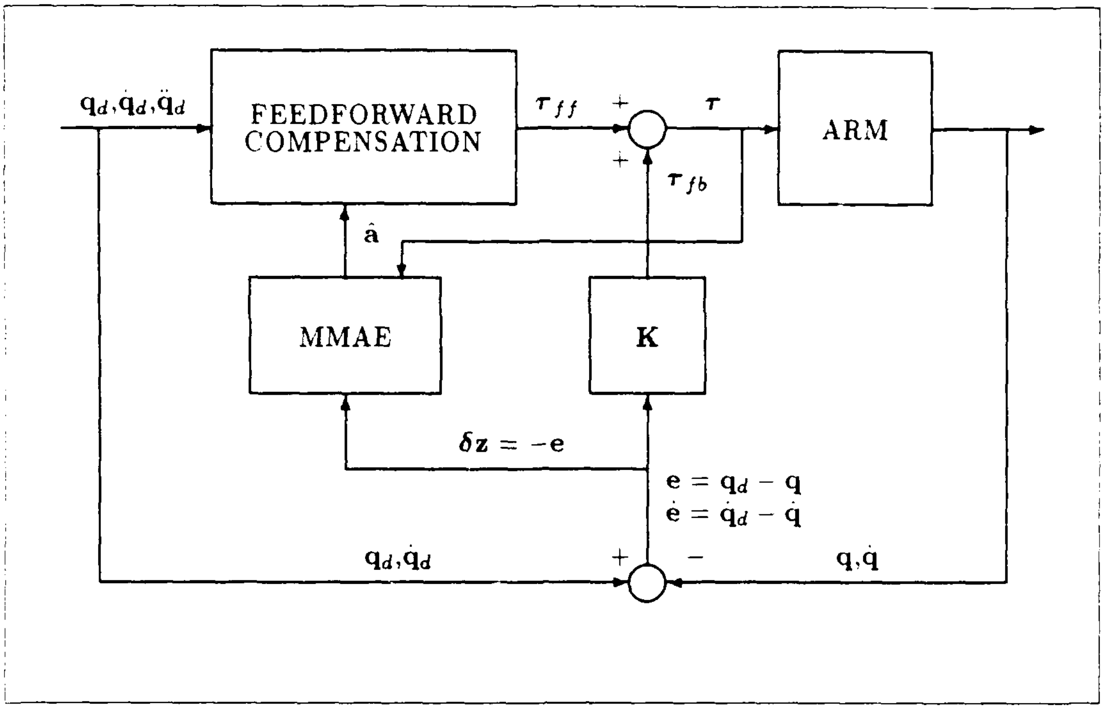


Figure 4.1. Open-Loop Multiple Model-Based Control Block Diagram

- m = the dimension of the measurement vector.

A block diagram of the structure of the OL/MMBC algorithm is shown in Figure 4.1.

4.5 Summary

This chapter developed an open-loop MMBC algorithm which no longer uses the PD feedback torques as the perturbation torques. This modification allows the MMAE technique to be applied to robotic manipulators where dependence on payload can be seen in the degradation of tracking performance but does not provide significant differences in the eigenvalues of the closed-loop linearized plant matrix. The open-loop MMBC algorithm was implemented for the first three links of a PUMA-560 robot manipulator and evaluated in simulation and experimentally. The details of that implementation and evaluation are discussed in the next chapter.

V. Open-Loop Multiple Model-Based Control Implementation and Evaluation

5.1 Introduction

The previous chapter presented the general algorithm development for the open-loop Multiple Model-Based Control (OL/MMBC) for robotic manipulators. This chapter presents the details of the OL/MMBC algorithm implementation and evaluation for the first three links of a PUMA-560 manipulator.

The PUMA-560 manipulator was selected as a test bed for the OL/MMBC algorithm because:

1. the trajectory tracking performance has been extensively evaluated.
2. the payload parameter vector can be reduced to mass only with minimal effect on tracking performance for the first three links, and
3. the Air Force Institute of Technology Robotics Systems Laboratory (ARSL) has the facilities to evaluate the algorithm experimentally if simulation tests are successful.

5.2 Implementation

As mentioned in the previous chapter, the OL/MMBC algorithm combines a Multiple Model Adaptive Estimation (MMAE) algorithm with a model-based controller as shown in Figure 4.1. To implement the OL/MMBC algorithm for a PUMA-560 manipulator the following was required:

1. the matrices associated with the linear perturbation model of Equation (4.1),
2. the linear perturbation measurement transformation matrix of Equation (3.23),
3. the Kalman filter equations, and
4. the PD feedback loop gains.

*5.2.1 **Perturbation Equations.*** The linear perturbation state equation shown in Equation (4.1) requires evaluation of the $\mathbf{F}'$, $\mathbf{B}$, and $\mathbf{G}$ matrices. The linearized plant matrix

$\mathbf{F}'$ for the first three links of the PUMA-560 can be written as:

$$\mathbf{F}'(\mathbf{x}_d, \boldsymbol{\tau}_d, a, t) = \left[\begin{array}{c} \mathbf{0}_{3\times3} \mid \mathbf{I}_{3\times3} \\ ------ \\ \left. \frac{\partial \mathbf{f}(\mathbf{x}, \boldsymbol{\tau}, a, t)}{\partial \mathbf{x}} \right|_d \end{array} \right]_{6\times6} \tag{5.1}$$

In order to generate Equation (5.1), the corrected MACSYMA batch routine originally developed by Tellman was used [30]. MACSYMA is a commercial software package that manipulates symbolic equations and develops FORTRAN software code [28]. The details of the lower partition of Equation (5.1) is contained in Appendix B and the listing of the MACSYMA batch routines are contained in [23].

The white Gaussian dynamics noise was assumed to enter the system at the same point as the perturbation torques. Therefore, $\mathbf{G}$ will equal $\mathbf{B}$ and can be written as:

$$\mathbf{G}(t) = \mathbf{B}(\mathbf{x}_d, \boldsymbol{\tau}_d, a, t) = \left[\begin{array}{c} \mathbf{0}_{3\times3} \\ ----- \\ \left[\mathbf{D}(\mathbf{q}, a, t) + \mathbf{N}^2\mathbf{J}_m\right]^{-1} \end{array} \right]_{6\times3} \tag{5.2}$$

The only measurements available on a PUMA-560 manipulator are joint positions. Therefore, the perturbation measurement transformation matrix can be written for all time as:

$$\mathbf{H}(t_i) = \left[\; \mathbf{I}_{3\times3} \mid \mathbf{0}_{3\times3} \; \right]_{3\times6} \tag{5.3}$$

5.2.2 Kalman Filter Equations. A total of three Kalman filters were used for this case study. Each of the filters was based on a different payload value. The three payload values were set to 0.0, 2.5, and 5.0 Kg. These payload values cover the the span of payload possibilities for the PUMA-560 and were the same payload values used during Tellman's research [30:4–14]. The Kalman filter equations were developed in Chapter 4 and are repeated here for further discussion:

$$\delta\hat{\mathbf{x}}_k(t_{i+1}^-) = \boldsymbol{\Phi}_k(t_{i+1}, t_i)\,\delta\hat{\mathbf{x}}_k(t_i^+) + \int_{t_i}^{t_{i+1}} \boldsymbol{\Phi}_k(t_{i+1}, \xi)\,\mathbf{B}_k(\mathbf{x}_d, \boldsymbol{\tau}_d, \mathbf{a}_k, \xi)\,\delta\boldsymbol{\tau}_k(\xi)d\xi \tag{5.4}$$

$$\mathbf{P}_k(t_{i+1}^-) = \boldsymbol{\Phi}_k(t_{i+1}, t_i)\,\mathbf{P}_k(t_i^+)\,\boldsymbol{\Phi}_k^T(t_{i+1}, t_i) + \mathbf{Q}_{d_k}(t_i) \tag{5.5}$$

$$\delta\hat{\mathbf{x}}_k(t_i^+) = \delta\hat{\mathbf{x}}_k(t_i^-) + \mathbf{K}_k(t_i)\left[\delta\mathbf{z}_i - \mathbf{H}_k(t_i)\,\delta\hat{\mathbf{x}}_k(t_i^-)\right] \tag{5.6}$$

$$\mathbf{P}_k(t_i^+) = \mathbf{P}_k(t_i^-) - \mathbf{K}_k(t_i)\mathbf{H}_k(t_i)\mathbf{P}_k(t_i^-) \tag{5.7}$$

$$\mathbf{K}_k(t_i) = \mathbf{P}_k(t_i^-)\mathbf{H}_k^T(t_i)\Big[\mathbf{H}_k(t_i)\mathbf{P}_k(t_i^-)\mathbf{H}_k^T(t_i) + \mathbf{R}_k(t_i)\Big]^{-1} \tag{5.8}$$

As was done by Tellman, the dynamics noises were assumed to be pair-wise independent and independent of the measurement noises. Furthermore, the dynamics noise strength $\mathbf{Q}_k$ and measurement noise covariance $\mathbf{R}_k$ were assumed to be constant throughout the trajectory.

To reduce the on-line computation time required to calculate the Kalman filter equations and parameter estimate, the linearized plant matrix ($\mathbf{F}'$), state transition matrix ($\boldsymbol{\Phi}$), covariance matrices ($\mathbf{P}(t_i^+)$ and $\mathbf{P}(t_i^-)$), and Kalman filter gains ($\mathbf{K}(t_i)$) were precomputed. In order to precompute the $\mathbf{F}'$ matrix, the nominally desired trajectory and torques were required. The desired trajectory was generated using a minimum-jerk generator, and the nominal torques were generated as shown in Equation (5.13).

The state transition matrix is a function of $\mathbf{F}'$ which varies with time. Assuming $\mathbf{F}'$ varies slowly with respect to the sample period the state transition matrix can be approximated as:[19:357]

$$\boldsymbol{\Phi}_k(t_{i+1}, t_i) \approx \mathbf{I} + \mathbf{F}'(\mathbf{x}_d, \boldsymbol{\tau}_d, a_k, t_i)\Delta t + \frac{1}{2}\left[\mathbf{F}'(\mathbf{x}_d, \boldsymbol{\tau}_d, a_k, t_i)\right]^2 \Delta t^2 \tag{5.9}$$

where Δt is the sample period. Similarly with $\mathbf{Q}_k$ held constant, $\mathbf{Q}_{d_k}(t_i)$ can be approximated as: [19:357]

$$\mathbf{Q}_{d_k}(t_i) \approx \mathbf{G}_k(t_i)\,\mathbf{Q}_k\,\mathbf{G}_k^T(t_i)\,\Delta t \tag{5.10}$$

Furthermore, the integration term on the right hand side of Equation (5.4) was approximated as:

$$\int_{t_i}^{t_{i+1}} \boldsymbol{\Phi}_k(t_{i+1}, \xi)\,\mathbf{B}_k(\mathbf{a}_k, \xi)\,\delta\boldsymbol{\tau}_k(\xi)d\xi \approx \boldsymbol{\Phi}_k(t_{i+1}, t_i)\,\mathbf{B}_k(\mathbf{a}_k, t_i)\,\delta\boldsymbol{\tau}_k(t_i)\,\Delta t \tag{5.11}$$

where:

$$\delta\boldsymbol{\tau}(t_i) = \boldsymbol{\tau}(t_i) - \boldsymbol{\tau}_{d_k}(t_i) \tag{5.12}$$

$$\boldsymbol{\tau}_{d_k}(t_i) = [\hat{\mathbf{D}}(\mathbf{q}_d, \mathbf{a}_k) + \mathbf{N}^2\mathbf{J}_m]\ddot{\mathbf{q}} + \hat{\mathbf{h}}(\dot{\mathbf{q}}_d, \mathbf{q}_d, \mathbf{a}_k) + \mathbf{N}^2\mathbf{B}_m\dot{\mathbf{q}}_d + \boldsymbol{\tau}_s + \hat{\mathbf{g}}(\mathbf{q}_d, \mathbf{a}_k) \tag{5.13}$$

Since the PUMA-560 starts from a known initial condition, the initial perturbation errors were assumed to be zero with a probability of one. This assumption resulted in starting the Kalman filter perturbation state estimates from zero ($\delta\hat{\mathbf{x}}_k(t_0) = \mathbf{0}$) with an initial covariance of zero ($\mathbf{P}_k(t_0) = \mathbf{0}$).

Since the payload parameter was limited to mass only, an explicit point mass calculation was used to determine the overall (link 3/payload) mass, centroids, and radii of gyrations. Assuming the external payload is a point mass rigidly attached to the last three links of the PUMA-560, the explicit point mass calculations can be written as:

$$m_3 = 6.97 + m_{ext} \tag{5.14}$$

$$\bar{x}_3 = \frac{6.97(0.01466)}{m_3} \tag{5.15}$$

$$\bar{y}_3 = \frac{6.97(0.00845)}{m_3} \tag{5.16}$$

$$\bar{z}_3 = \frac{6.97(0.1101) + m_{ext}(0.48935)}{m_3} \tag{5.17}$$

$$k^2_{3_{xx}} = k^2_{3_{yy}} = \frac{0.0783(6.97) + m_{ext}(0.48935)^2}{m_3} \tag{5.18}$$

$$k^2_{3_{zz}} = \frac{0.0021(6.97)}{m_3} \tag{5.19}$$

where:

- m_3 = the overall mass of the third link and external payload in kilograms.
- m_{ext} = the external mass in kilograms.
- $\bar{x}_3$ = x-axis mass centroid location in meters.
- $\bar{y}_3$ = y-axis mass centroid location in meters.
- $\bar{z}_3$ = z-axis mass centroid location in meters.
- $k^2_{3_{xx}}$ = radius of gyration about the x-axis.
- $k^2_{3_{yy}}$ = radius of gyration about the y-axis.
- $k^2_{3_{zz}}$ = radius of gyration about the z-axis.

Table 5.1. PD Feedback Gains

Soft Gains		
Link i	Position ($K_{p_{ii}}$)	Velocity ($K_{v_{ii}}$)
1	250.0	72.0
2	520.0	129.0
3	95.6	24.8

High Gains		
Link i	Position ($K_{p_{ii}}$)	Velocity ($K_{v_{ii}}$)
1	640.0	72.0
2	1330.0	129.0
3	360.0	24.8

5.2.3 PD Feedback Gains. The PD feedback loop of the model-based controller uses a set of position ($\mathbf{K_p}$) and velocity ($\mathbf{K_v}$) gains. Two sets of gains were used as part of this evaluation and are tabulated in Table 5.1. The first set was the same as those used during Tellman's research [30:4–12] and henceforth will be referred to as soft gains. The second set was a result of recent research efforts [12] to improve tracking performance and will be denoted as high gains.

5.2.4 Software. The software used to implement the OL/MMBC algorithm was based on the software developed by Tellman. Modifications were made to the software to account for the differences between the CL/MMBC and OL/MMBC algorithms. A complete listing of the software used during this research can be found in [23].

5.3 Simulation Evaluation

The PUMA-560 manipulator motion was simulated using a digital simulation. This digital simulation used a fourth order Runge-Kutta integration scheme with 1 millisecond subintervals to solve Equation (3.3). A block diagram of the simulator and OL/MMBC algorithm is shown in Figure 5.1.

This simulator is a modified version of the simulator used by Tellman [30:4–14] which allows for selection of sample periods that are multiples of 0.9 ms. This modification was a

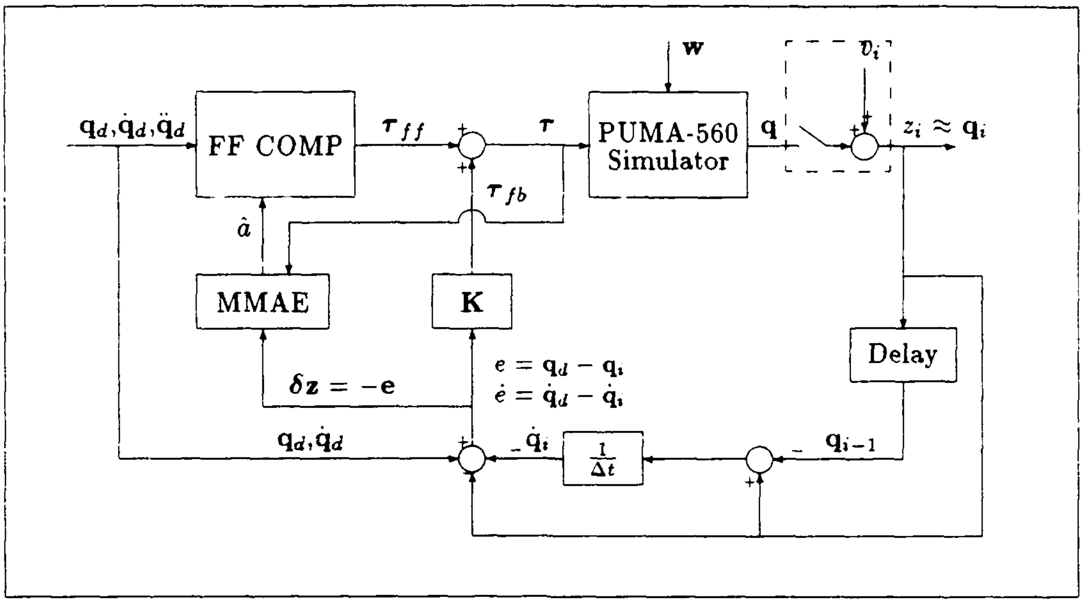

w
v_i
$\mathbf{q}_d, \dot{\mathbf{q}}_d, \ddot{\mathbf{q}}_d$
FF COMP
τ_{ff}
+
+
τ
τ_{fb}
PUMA-560 Simulator
q
+
+
$z_i \approx \mathbf{q}_i$
$\hat{a}$
MMAE
K
Delay
$\delta\mathbf{z} = -\mathbf{e}$
$e = \mathbf{q}_d - \mathbf{q}_i$
$\dot{e} = \dot{\mathbf{q}}_d - \dot{\mathbf{q}}_i$
$\mathbf{q}_d, \dot{\mathbf{q}}_d$
+
$-\dot{\mathbf{q}}_i$
$\frac{1}{\Delta t}$
$-\ \mathbf{q}_{i-1}$
+

Figure 5.1. Block Diagram of OL/MMBC Simulator

result of changes made to the timing source used for experimental evaluations and will be discussed later. The sample period used for the simulation evaluation was 7.2 ms, which is close to the 7.0 ms simulation sample period used by Tellman [30].

5.3.1 Filter Tuning Procedures. Since the model-based controller provides good high speed tracking performance when the payload is known, the Kalman filter noises of the MMAE algorithm were tuned ιo achieve the best possible payload parameter estimate. The tuning process was accomplished using a minimum-jerk trajectory profile that moved the first three joints of the manipulator from (−50°, −135°, 135°) to (50°, −85°, 30°) in 2.0 seconds. The trajectory profile is denoted as Trajectory One and is shown in Figure A.1.

During the tuning process, two hypothesis conditional probability computation approaches were used to tune the filters. The first approach calculated the hypothesis conditional probabilities as shown in Equation (4.17). The second approach reset $p_k(t_{i-1})$ of Equation (4.17) to 1/3 for all time. The second approach results in discarding all previous measurements except for the current measurement to come up with the probability calculation at t_i. This second approach resulted from Tellman's research [30:4-43] of the CL/MMBC. In both approaches, the filters were tuned for three different external payloads. These external payloads corresponded, respectively, to the three filter assumed payloads (0.0, 2.5, and 5.0 Kg).

For each external payload and approach, the **R** values were held constant while the **Q** values were decreased by orders of magnitude for each simulation run. The payload estimate and probabilities of each filter were recorded and plots of this data are contained in Appendix C. Table 5.2 shows the noise levels for each filter that provided the best payload estimation performance for each payload. As Table 5.2 shows, the first approach would require the use of different noise values depending on the external payload; however, the second approach used the same noise levels for each external payload. Since the noises were assumed to remain constant throughout the trajectory, the second approach was used for the simulation evaluations.

Table 5.2. Kalman Filter Noise Levels For Best Performance

$p_k(t_{i-1})$ not reset to 1/3							
External Payload (Kg)	Filter No.	Q_{11}	Q_{22}	Q_{33}	R_{11}	R_{22}	R_{33}
5.0	1,2, and 3	10.0	10.0	10.0	10^{-6}	10^{-6}	10^{-6}
2.5	1,2, and 3	1.0	1.0	1.0	10^{-6}	10^{-6}	10^{-6}
0.0	1,2, and 3	1.0	1.0	1.0	10^{-6}	10^{-6}	10^{-6}
$p_k(t_{i-1})$ reset to 1/3							
External Payload (Kg)	Filter No.	Q_{11}	Q_{22}	Q_{33}	R_{11}	R_{22}	R_{33}
5.0	1,2, and 3	0.01	0.01	0.01	10^{-6}	10^{-6}	10^{-6}
2.5	1,2, and 3	0.01	0.01	0.01	10^{-6}	10^{-6}	10^{-6}
0.0	1,2, and 3	0.01	0.01	0.01	10^{-6}	10^{-6}	10^{-6}

Using the noise values of the second approach and the soft gains in Table 5.1, the OL/MMBC algorithm was evaluated in simulation for Trajectory One with external payloads of 0.0, 1.0, 2.0, 2.5, 3.0, 4.0, and 5.0 Kg. For each simulation run, the payload estimate, hypothesis conditional probabilities, position tracking error, and commanded torques data were recorded. Additionally, simulations were conducted for a single model-based control (SMBC) algorithm with and without knowledge of the payload information.

These SMBC simulations were used as a benchmark for tracking performance comparisons with the OL/MMBC.

5.3.2 Results. Plots of the payload estimates, probabilities, and tracking error plots for the payloads simulated are contained in Appendix C.

5.3.2.1 Payload Estimate Performance. The payload estimate provided by the MMAE reached a steady-state value within the first 0.3 seconds of the trajectory when the external payload matched one of the filter's assumed payload value. The steady-state estimate correctly identified the actual external payload. This is illustrated in Figure 5.2 for the 5.0 Kg payload. The corresponding probability plots of the three filters are shown in Figure 5.3. As Figure 5.3 shows, the probability associated with the filter based on a =5.0 Kg quickly approaches one. Similar results were obtained for external payloads of 2.5 and 0.0 Kg (see Appendix C).

However, when the external payload differed from one of the filter's assumed payload value, the results showed that the payload estimates varied. For external payloads of 4.0 and 1.0 Kg, the MMAE payload estimates oscillated as shown in Figures C.34 and C.46 for the two respective payloads. Figure C.34 shows that the payload estimate oscillated between 5.0 and 2.5 Kg in a square-wave like pattern. These estimates correspond to the filters based on a = 5.0 and a = 2.5. Similarly the payload estimate for the 1.0 Kg payload oscillated between 2.5 and 0.0 Kg and corresponds to the filters based on a = 2.5 and a = 0.0. These results indicate that the MMAE has difficulty estimating the actual payload as it approaches a point about half way between the filters. The MMAE payload estimates for 3.0 and 2.0 Kg locked onto a value 2.5 Kg within the first 0.2 seconds of the trajectory as shown in Figures C.37 and C.43. This estimate corresponds to the filter based on a = 2.5 Kg which happens to be the closest of the three filters to the actual payloads.

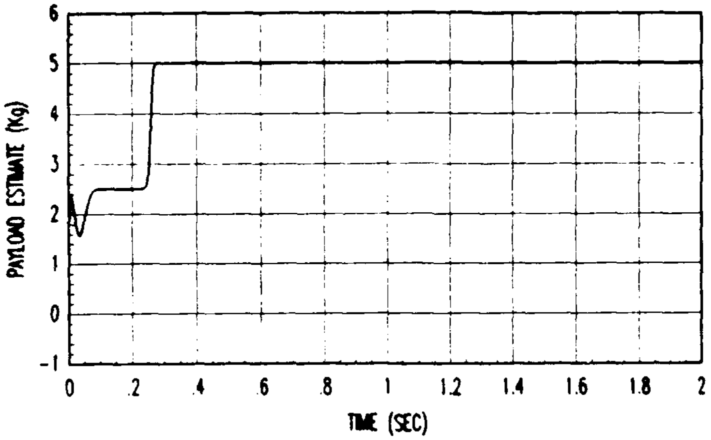


Figure 5.2. MMAE Payload Estimate for External Payload of 5.0 Kg

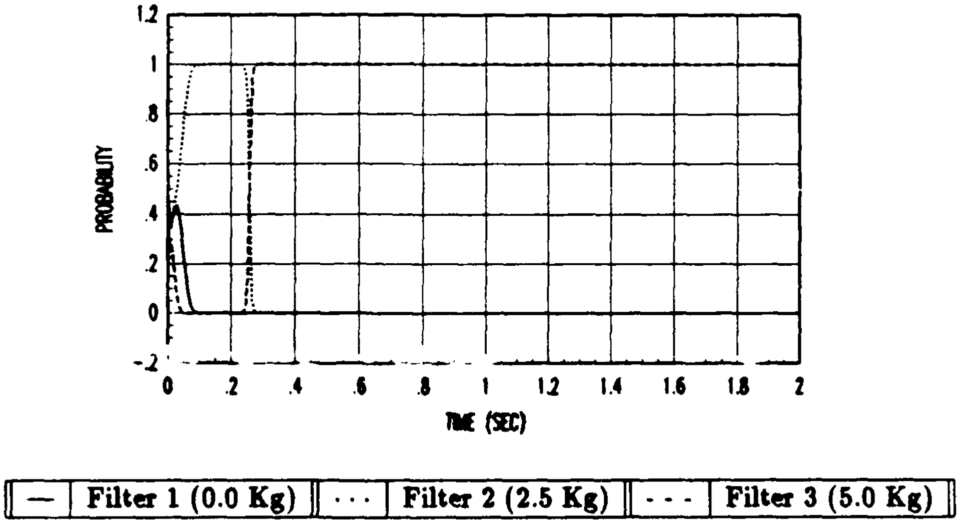


Figure 5.3. Hypothesis Conditional Probabilities for 5.0 Kg External Payload

5.3.2.2 Tracking Performance. The tracking performance of any algorithm indicates how well it performs in comparison to other algorithms. Therefore, the OL/MMBC was compared to the tracking performance of an SMBC with and without knowledge of the payload. The tracking ability of an informed SMBC is the best that the OL/MMBC could expect to achieve. The tracking errors for an external payload of 5.0 Kg are shown in Figure 5.4. As Figure 5.4 shows, the tracking errors are comparable to the artificially informed SMBC and significantly better than the uninformed SMBC tracking errors. However, the peak errors of the OL/MMBC are slightly higher than the informed SMBC and can be related to the initial payload estimate being incorrect at the start of the trajectory (see Figure 5.2). Similar performance was obtained for the 2.5 and 0.0 Kg payload cases (see Figures C.42 and C.51).

Figures C.36 and C.48 show the tracking performance for the 4.0 and 1.0 Kg payloads respectively. Figure C.36 shows that although the payload estimate was poor, the tracking performance was still better than the uninformed SMBC. However, the poor payload estimate for the 1.0 Kg payload resulted in poor tracking performance. The final errors for joints 2 and 3 were comparable to the informed SMBC; however the peak errors were on the same order of magnitude as the uninformed SMBC.

Even for the 3.0 and 2.0 Kg payloads where the estimate differed by 0.5 Kg, the tracking error was not drastically different from the informed SMBC and was significantly better than the uninformed SMBC. A summary of the peak and final position errors for the OL/MMBC, informed SMBC, and uninformed SMBC is contained in Table 5.3

5.3.3 Discussion The payload estimation for those simulated payloads raised some issues about the parameter discretization. The payload parameter was divided into three evenly spaced discrete values over the continuous payload range from 0.0 to 5.0 Kg. This resulted in payload parameter values of 0.0, 2.5, and 5.0 Kg. The results indicated that when the payload was about halfway between 0.0 and 2.5 or 2.5 and 5.0 the MMAE had difficulty resolving a good payload estimate. This poor payload estimate resulted in poor tracking performance. One possible cause for this is that the parameter discretization used may not have been the optimum choice, or a finer discretization is required.

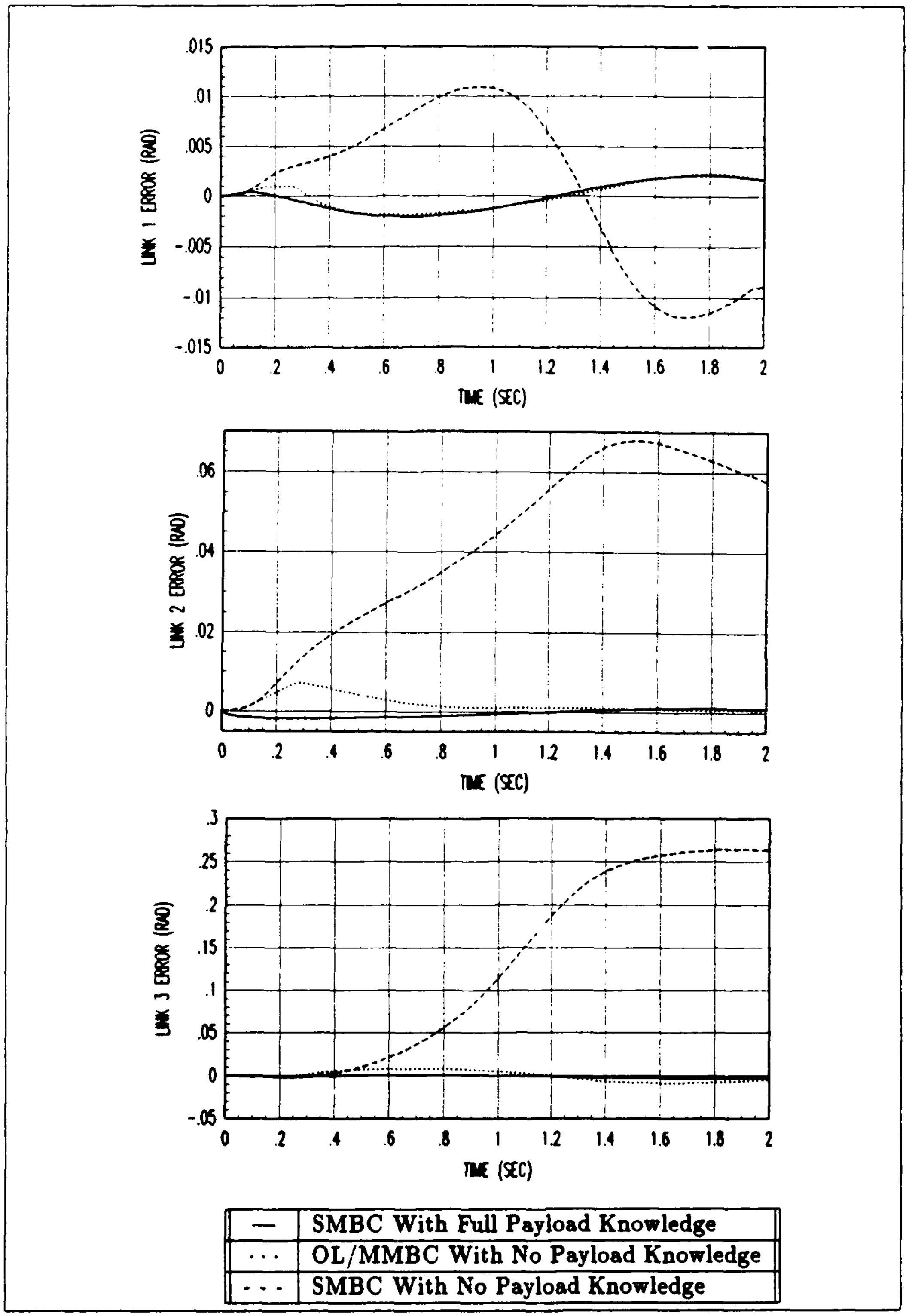


Figure 5.4. Payload 5.0 Kg: Trajectory One Tracking Error

Table 5.3. Peak and Final Position Errors for Trajectory One

OL/MMBC						
External	Peak Error (rad)			Final Error (rad)		
Payload (Kg)	Joint 1	Joint 2	Joint 3	Joint 1	Joint 2	Joint 3
5.0	0.0023	0.0070	0.0087	0.0018	0.0005	0.0049
4.0	0.0039	0.0109	0.0032	0.0004	0.0013	0.0023
3.0	0.0012	0.0068	0.0220	0.0009	0.0062	0.0220
2.5	0.0022	0.0009	0.0059	0.0017	0.0007	0.0032
2.0	0.0032	0.0051	0.0287	0.0025	0.0044	0.0276
1.0	0.0045	0.0079	0.0407	0.0022	0.0009	0.0080
0.0	0.0022	0.0066	0.0033	0.0017	0.0008	0.0022

SMBC with Payload Information						
External	Peak Error (rad)			Final Error (rad)		
Payload (Kg)	Joint 1	Joint 2	Joint 3	Joint 1	Joint 2	Joint 3
5.0	0.0021	0.0018	0.0030	0.0017	0.0008	0.0022
4.0	0.0021	0.0012	0.0028	0.0017	0.0008	0.0021
3.0	0.0021	0.0011	0.0026	0.0017	0.0008	0.0020
2.5	0.0021	0.0011	0.0026	0.0017	0.0008	0.0020
2.0	0.0021	0.0011	0.0025	0.0017	0.0008	0.0019
1.0	0.0021	0.0011	0.0024	0.0017	0.0008	0.0018
0.0	0.0021	0.0011	0.0023	0.0017	0.0008	0.0018

SMBC without Payload Information						
External	Peak Error (rad)			Final Error (rad)		
Payload (Kg)	Joint 1	Joint 2	Joint 3	Joint 1	Joint 2	Joint 3
5.0	0.0119	0.0678	0.2642	0.0089	0.0579	0.2636
4.0	0.0077	0.0506	0.2027	0.0059	0.0449	0.2027
3.0	0.0048	0.0364	0.1470	0.0035	0.0327	0.1470
2.5	0.0035	0.0298	0.1204	0.0024	0.0268	0.1203
2.0	0.0024	0.0236	0.0944	0.0014	0.0212	0.0944
1.0	0.0007	0.0118	0.0447	0.0003	0.0106	0.0447
0.0	This is the assumed no payload					

If the payload parameter space requires a finer discretization, then more filters would be required. These additional filters will increase the computation time. Furthermore, for a different manipulator that can carry more payload than 5.0 Kg, even more filters would be required. This is where the CL/MMBC algorithm developed by Tellman has an advantage. The CL/MMBC estimates payload variations and uses the estimate of the payload variation to get a payload estimate. Therefore, a fewer bank of filters can be used to estimate a wider range of payloads [30,15]. Unfortunately, Chapter 3 discovered limitations for manipulators like the PUMA-560 whose performance degrades when payload information is not known but whose linearized plant models showed little dependence on the payload parameter.

Another point of concern is the process of tuning the filters to achieve good payload performance. The ad hoc approach used during this evaluation obtained good results for those payload cases that matched one of the filters payload value. These may not have been the best values to use and the experimental evaluation discussed later required some retuning. The performance of the MMAE algorithm can be effected by poor selection of the filter noises [9,20].

5.4 Experimental Evaluation

The experimental evaluations performed in this research effort were accomplished using the AFIT Robotic Control Algorithm Development and Evaluation (ARCADE) system [13]. The ARCADE system allows for user friendly evaluation of different forms of position control algorithms.

5.4.1 Test Environment. ARCADE is hosted on a VAXstation III with parallel and serial connections to the PUMA's LSI-11/73 computer bus. The LSI-11/73 computer serves as a preprocessor by:[12]

- sending control codes to the PUMA digital servo boards to command motor currents and read encoder values,
- converting encoder counts to angular position, and

- communicating with the VAXstation every sample period.

The DRV11-J parallel interface device is used to pass the joint positions and motor currents between the LSI-11/73 computer and the VAXstation. The DRV11-J is driven by software supplied by the VAXlab software package [3]. The VAXstation uses the VMS operating system and requires 2.25 ms to establish the communication link between the LSI-11/73 and the VAXstation.

5.4.1.1 Timing. The PUMA electronics provide a nominal 7ms clock pulse to the BEVENT line of the LSI-11/73 controller bus [32]. This timing source is used by VAL-II to control the PUMA motion and was the timing source used during Tellman's research. This timing source restricted the sample rates to multiples of 7ms [14,30]. A new timing source was created to allow for higher sample rates afforded by the VAXstation computation power. The new timing source used a KW11-C clock card installed on the VAXstation Q-bus. The KW11-C is triggered by a 1MHz clock provided by the PUMA controller and replaces the 7ms clock pulse used by the BEVENT line [12]. The clock rate is user selectable at 0.9ms increments. The 0.9ms increments are a result of the PUMA optical hardware restrictions. The encoder counters are clocked at the servo rate of the original VAL-II controller. The nominal servo rate stated by Unimation is 0.875 ms; however the actual encoder update rate of the electronics used by this research is 0.9 ms.

5.4.2 Procedures. The OL/MMBC algorithm was evaluated using Trajectory One and external payloads of 3.0, 2.0, 1.0, and 0.0 Kg. These payloads were simulated and the experimental evaluations will validate the simulation results.

Initial attempts to evaluate the OL/MMBC algorithm at the 7.2 ms sample period were unsuccessful due to the computation time required of the OL/MMBC algorithm. Incrementing the sample period by 0.9 ms increments eventually led to a sample period of 9.9 ms. This sample period was used for all subsequent experimental tests.

Using the filter noise levels obtained during simulations, initial experimental runs were conducted for both payloads. These runs indicated that the noise levels needed to be retuned. The same tuning process described in Section 5.3.1 was used. Efforts at retuning

Table 5.4. Filter Noise Levels used in Experimental Evaluation

Filter	Q_{11}	Q_{22}	Q_{33}	R_{11}	R_{22}	R_{33}
1 (0.0 Kg)	0.08	0.08	0.08	10^{-6}	10^{-6}	10^{-6}
2 (2.5 Kg)	0.07	0.07	0.07	10^{-6}	10^{-6}	10^{-6}
3 (5.0 Kg)	0.06	0.06	0.06	10^{-6}	10^{-6}	10^{-6}

were unsuccessful, and an analysis of the commanded torques versus feedforward torques of a SMBC artificially informed of the 3.0 Kg payload information showed that the commanded torques differed from the feedforward torques (see Figure 5.5). This observation was different from the simulation results obtained in Chapter 3 which showed no difference between the commanded and feedforward torques. The difference between the torques was not unexpected since certain drive system dynamics were not modeled. However, the difference between the commanded and feedforward torques could cause significant perturbations and invalidate the linearized modeling assumption. Therefore, a modification to the OL/MMBC algorithm was made. This modification changed the perturbation torque calculation of Equation (5.12) from the difference between the commanded torque and the nominal torque to the difference between the actual feedforward torque and the nominal torque of the filter and can be written as:

$$\delta\boldsymbol{\tau}(t_i) = \boldsymbol{\tau}_{ff}(t_i) - \boldsymbol{\tau}_{d_k}(t_i) \qquad (5.20)$$

where:

- $\boldsymbol{\tau}_{ff}(t_i)$ = the feedforward torque calculated at time t_i.
- $\boldsymbol{\tau}_{d_k}(t_i)$ = the nominal torque for the kth filter as calculated in Equation (5.13).

Using this modification, the filters were retuned as described earlier for an external payload of 3.0 Kg. The tuning resulted in good tracking performance and payload estimation for the 3.0 Kg case using the noise values shown in Table 5.4.

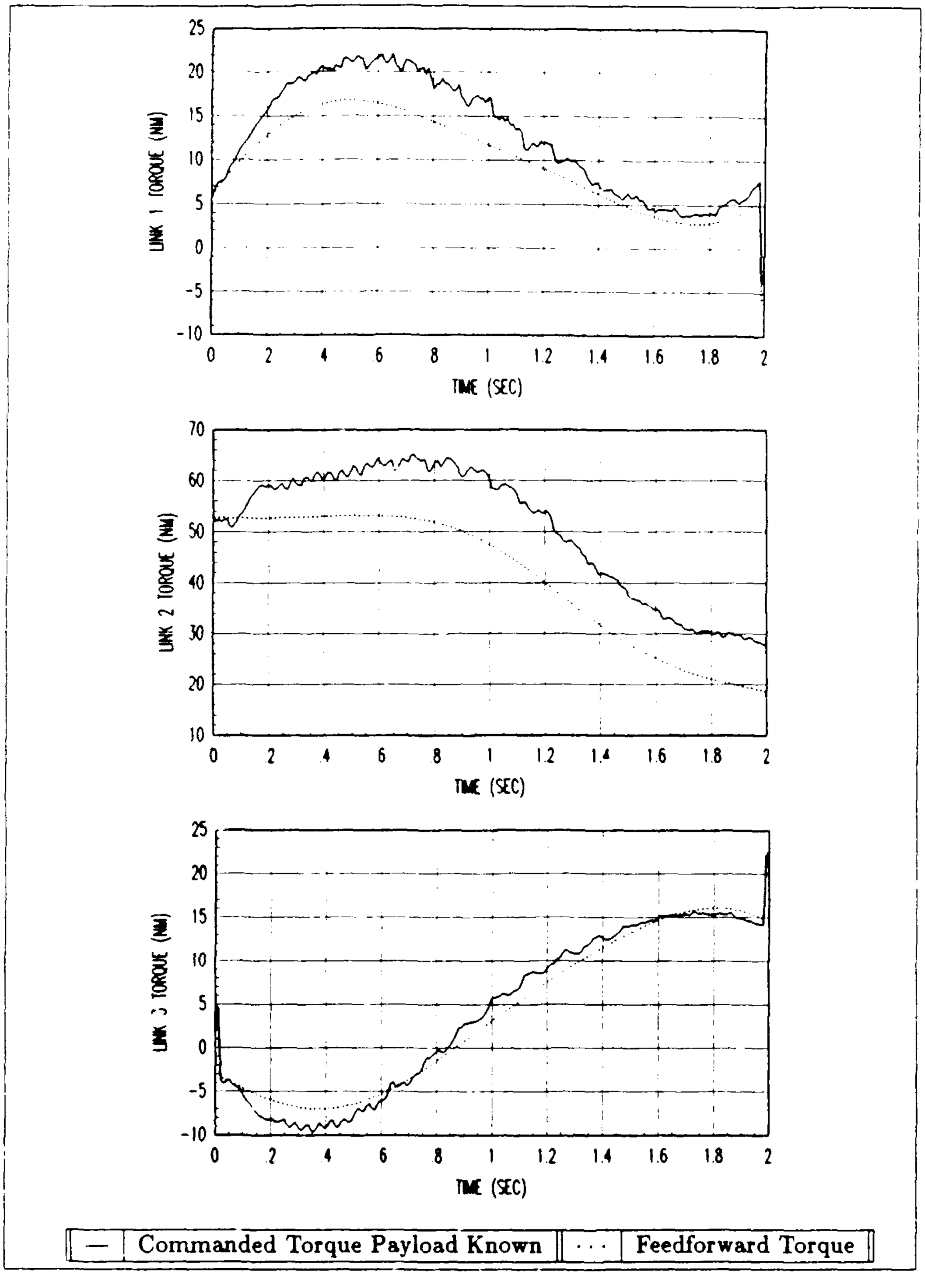


Figure 5.5. Trajectory One: Torque Comparison for External Payload of 3.0 Kg

Using the modification made to the OL/MMBC algorithm and noise levels shown in Table 5.4, the OL/MMBC was experimentally evaluated for Trajectory One using an external payload of 3.0, 2.0, 1.0, and 0.0 Kg. A total of 10 runs were accomplished for each payload and set of PD gains shown in Table 5.1.

As briefly mentioned earlier, the high PD gains were a result of recent experimental evaluations on the PUMA-560 manipulator . During those evaluations the use of these high gains improved the tracking errors of an informed SMBC [12]. The high gains were used in addition to th‹ soft gains to see what effects these tracking error improvements had on the operation of the OL/MMBC.

5.4.3 Results. Taking the ensemble of 10 runs for each payload and set of PD gains, the statistical mean and one sigma values were obtained. Plots of these statistics for the payload estimates and tracking errors are contained in Appendix D.

5.4.3.1 Payload Estimate Performance. For the payloads tested, the MMAE algorithm reached a steady-state estimate within the first 0.6 seconds of the trajectory. This steady-state performance was achieved for both sets of PD gains. The mean payload estimate for 3.0 Kg locked onto a value of 2.5 Kg and is shown in Figure 5.6 for the soft and high gains. Similar payload estimates for 2.0 Kg were also obtained and are shown in Appendix D. The mean payload estimate for the 0.0 Kg initially went in the wrong direction but locked onto a value of 0.0 Kg as shown in Figure 5.7. Note these payload estimates are similar to those estimates obtained in simulation and the PD gain values had no effect on the MMAE payload estimation performance. The 1.0 Kg payload estimate locked onto a value of 2.5 Kg, which is different from the results obtained in simulation. The differences could be due to the unmodeled drive system dynamics which required a modification to the the perturbation torque calculations and filter noise levels. The plot of the 1.0 Kg payload estimate is contained in Appendix D.

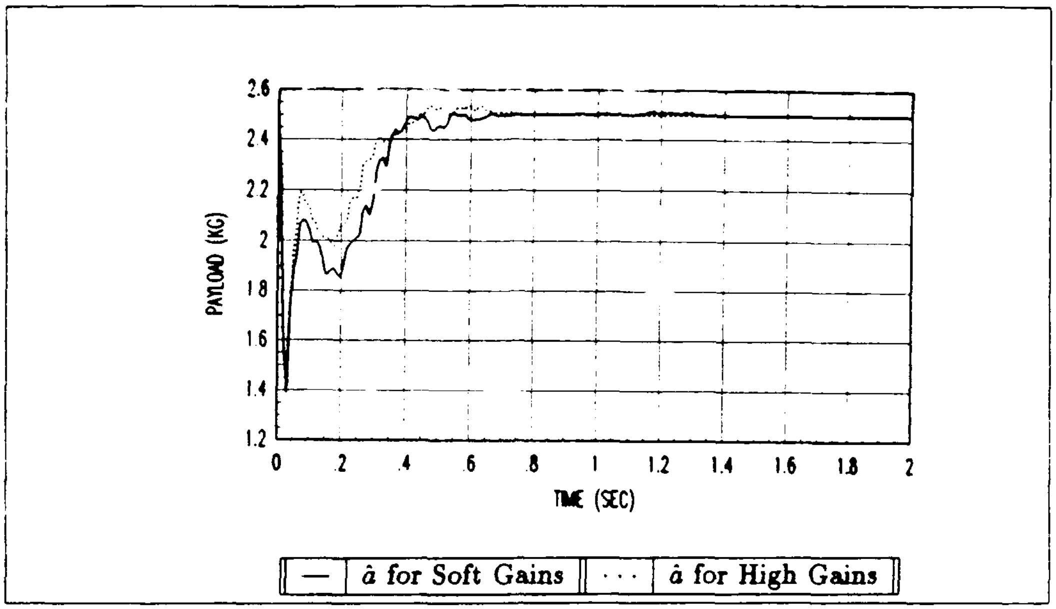


Figure 5.6. MMAE Estimate for External Payload of 3.0 Kg

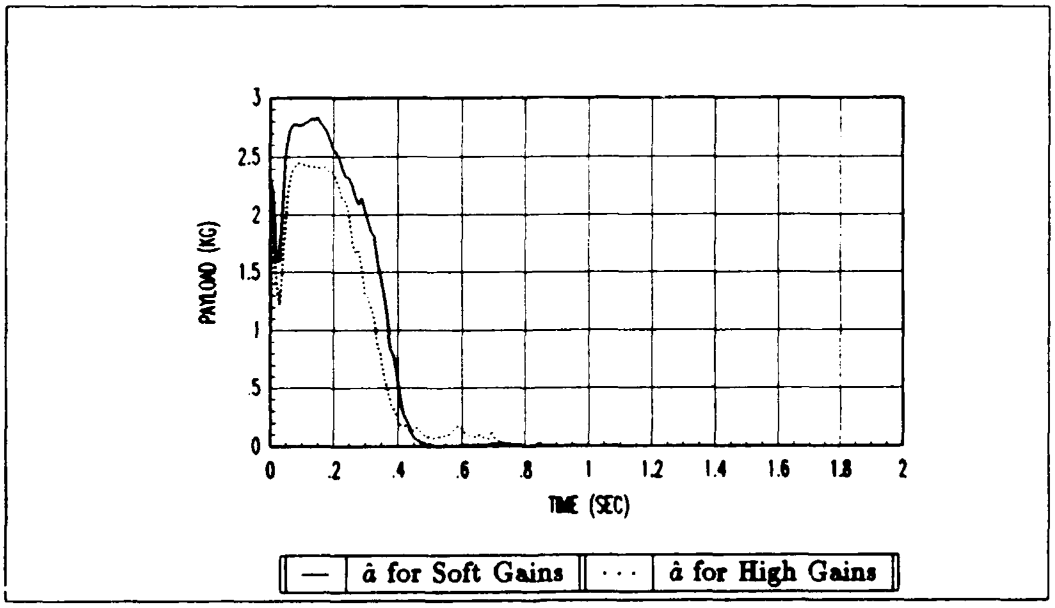


Figure 5.7. MMAE Estimate for External Payload of 0.0 Kg

5.4.3.2 Tracking Performance In order to compare the tı ıcking performance of the OL/MMBC, experimental runs of an informed and uninformed SMBC were conducted and the tracking errors recorded. A total of 10 runs were conducted for each payload case and the statistical mean of the 10 runs was used for comparisons. The mean tracking errors for the 3.0 Kg payload case are shown in Figures 5.8 and 5.9 for the soft and high gains respectively. As both figures show, the OL/MMBC tracking error performance was similar to the informed SMBC and significantly better than the uninformed SMBC. Also note that the tracking errors were smaller for the high gains. Similar profiles were obtained for the 2.0 and 0.0 Kg payloads and are shown in Appendix D. The 1.0 Kg tracking error plots are also shown in Appendix D. These tracking errors show that for this particular trajectory and starting conditions, the OL/MMBC end point errors for joint 3 were actually worse than the uninformed SMBC. This was not unexpected since the third link is influenced by the gravity field and the payload estimate was too large. However, the incorrect payload estimate improved the tracking error of the OL/MMBC for joint 2 when compared to the informed SMBC tracking error. To summarize the tracking errors, Tables 5.5 and 5.6 contain the mean peak and final errors (absolute values) for the OL/MMBC, informed, and uninformed SMBC for the soft and high gains, respectively.

5.5 Discussion

The initial simulation and experimental evaluation of the OL/MMBC on a PUMA-560 demonstrated its ability to adapt quickly to unknown payloads and provide adequate tracking performance. However, for certain payloads, the OL/MMBC was not able to provide a good payload estimate indicating that the parameter discretization was not optimum or a finer discretization was required. A finer discretization would require more filters and increase the computation time. The increased computation time can be improved by using paralle processing.

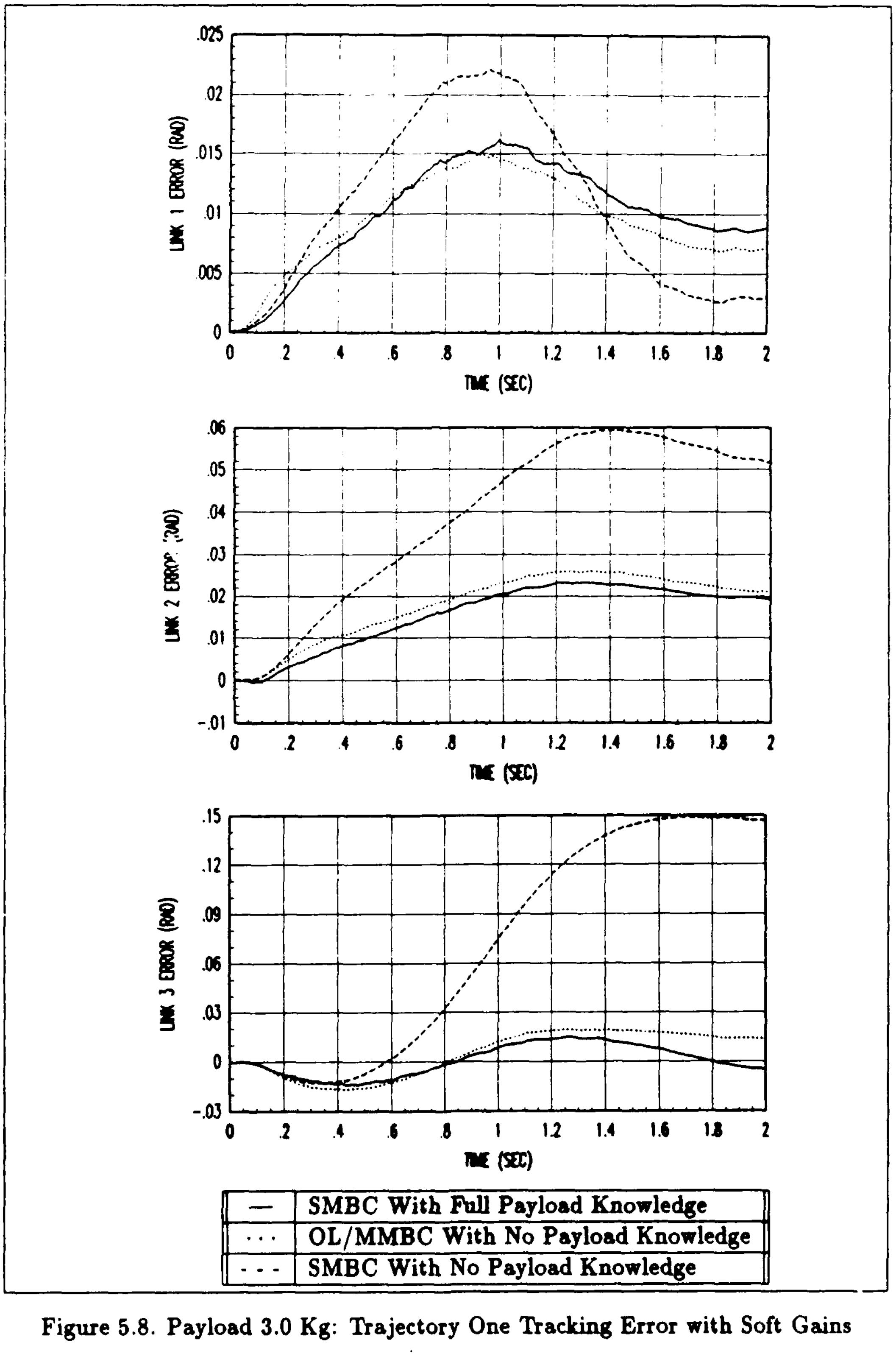


Figure 5.8. Payload 3.0 Kg: Trajectory One Tracking Error with Soft Gains

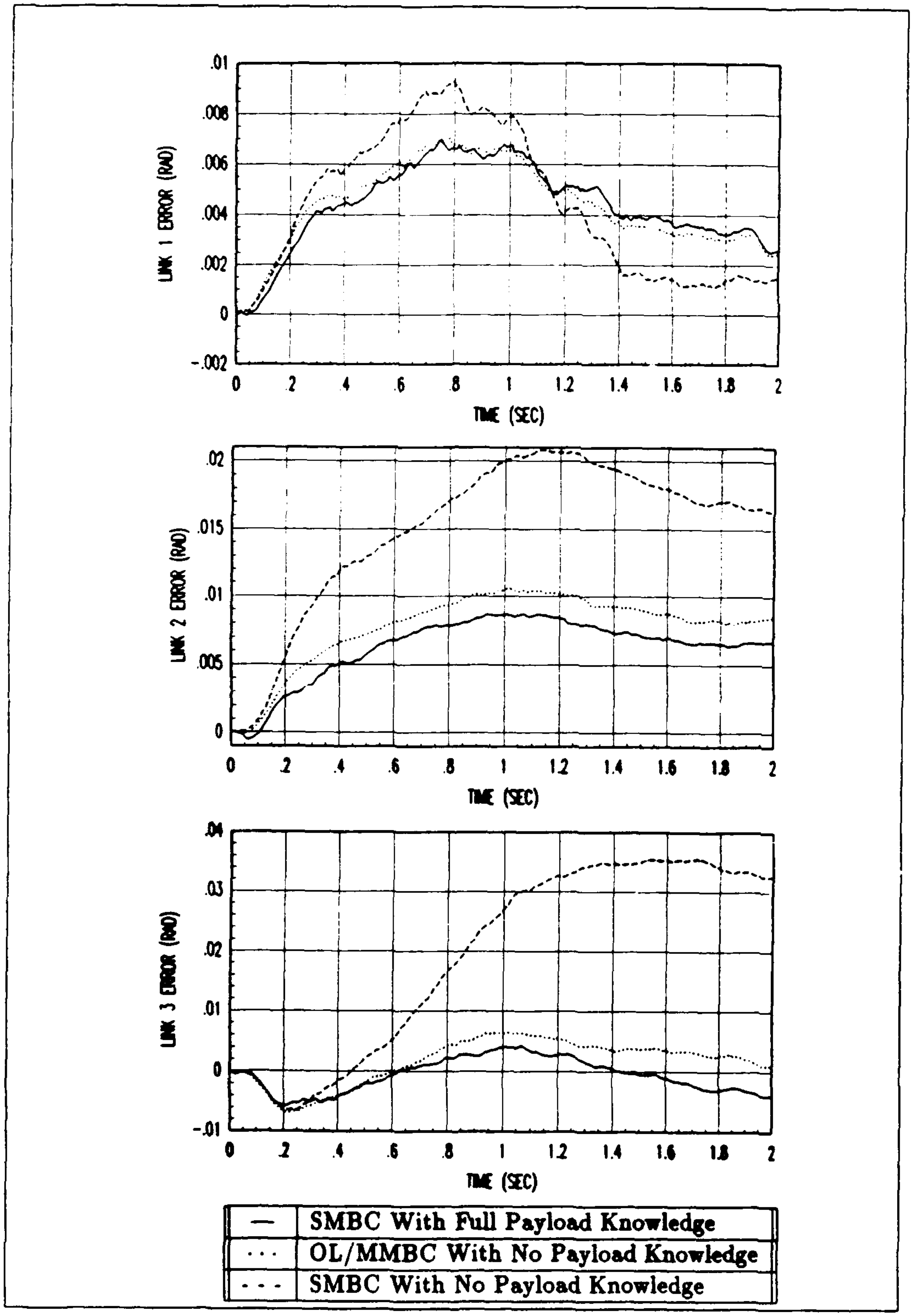


Figure 5.9. Payload 3.0 Kg: Trajectory One Tracking Error with High Gains

Table 5.5. Experimental Peak and Final Tracking Errors with Soft Gains

OL/MMBC						
External	Peak Error (rad)			Final Error (rad)		
Payload (Kg)	Joint 1	Joint 2	Joint 3	Joint 1	Joint 2	Joint 3
3.0	0.0149	0.0260	0.0191	0.0072	0.0211	0.0134
2.0	0.0102	0.0177	0.0295	0.0067	0.0138	0.0295
1.0	0.0101	0.0066	0.0747	0.0090	0.0009	0.0747
0.0	0.0158	0.0166	0.0089	0.0097	0.0145	0.0049

SMBC with Payload Information						
External	Peak Error (rad)			Final Error (rad)		
Payload (Kg)	Joint 1	Joint 2	Joint 3	Joint 1	Joint 2	Joint 3
3.0	0.0162	0.0232	0.0146	0.0089	0.0193	0.0049
2.0	0.0125	0.0194	0.0174	0.0061	0.0149	0.0006
1.0	0.0131	0.0164	0.0175	0.0066	0.0125	0.0010
0.0	0.0160	0.0170	0.0104	0.0097	0.0145	0.0043

SMBC without Payload Information						
External	Peak Error (rad)			Final Error (rad)		
Payload (Kg)	Joint 1	Joint 2	Joint 3	Joint 1	Joint 2	Joint 3
3.0	0.0221	0.0594	0.1489	0.0030	0.0516	0.1465
2.0	0.0173	0.0414	0.0992	0.0035	0.0351	0.0970
1.0	0.0149	0.0266	0.0542	0.0045	0.0222	0.0474
0.0	This is the assumed no payload info case					

Table 5.6. Experimental Peak and Final Tracking Errors with High Gains

OL/MMBC						
External	Peak Error (rad)			Final Error (rad)		
Payload (Kg)	Joint 1	Joint 2	Joint 3	Joint 1	Joint 2	Joint 3
3.0	0.0071	0.0106	0.0064	0.0025	0.0084	0.0010
2.0	0.0049	0.0067	0.0087	0.0020	0.0048	0.0087
1.0	0.0046	0.0028	0.0205	0.0023	0.0009	0.0205
0.0	0.0068	0.0069	0.0034	0.0027	0.0053	0.0028

SMBC with Payload Information						
External	Peak Error (rad)			Final Error (rad)		
Payload (Kg)	Joint 1	Joint 2	Joint 3	Joint 1	Joint 2	Joint 3
3.0	0.0070	0.0087	0.0042	0.0027	0.0067	0.0041
2.0	0.0056	0.0072	0.0052	0.0025	0.0053	0.0013
1.0	0.0056	0.0063	0.0055	0.0023	0.0043	0.0007
0.0	0.0068	0.0070	0.0038	0.0027	0.0053	0.0027

SMBC without Payload Information						
External	Peak Error (rad)			Final Error (rad)		
Payload (Kg)	Joint 1	Joint 2	Joint 3	Joint 1	Joint 2	Joint 3
3.0	0.0093	0.0208	0.0354	0.0016	0.0162	0.0324
2.0	0.0069	0.0149	0.0252	0.0017	0.0119	0.0212
1.0	0.0062	0.0099	0.0139	0.0021	0.0076	0.0121
0.0	This is the assumed no payload info case					

Also, the OL/MMBC required a modification to the perturbation torques for the experimental evaluations as shown in Equation (5.20). This modification was necessary since unmodeled drive system dynamics caused differences between the commanded and feedforward torques when the payload was known. The modification provided excellent performance for the initial evaluations after some filter tuning. Further analysis of the modification and its effects are warranted.

As mentioned earlier the tuning was accomplished using an ad hoc approach to achieve the best payload performance. The approach does not guarantee that the noises obtained will be the optimum values for all payload cases, and the noises were different between simulation and experimentation. There seems to be no systematic way of determining *a priori* what noise levels will provide the best performance of the filters. The ad hoc approach is not limited to the MMAE adaptation scheme alone. The other adaptation schemes reviewed in Chapter 2 also use ad hoc approaches to tune their adaption mechanism.

5.6 Summary

The results of initial simulation and experimental evaluations of the OL/MMBC algorithm for a PUMA-560 manipulator were presented. The results indicated that for certain payloads the MMAE algorithm could adapt quickly to the unknown payload and provide a good payload estimate. The tracking performance for those payloads was comparable to an artificially informed SMBC and significantly better than an uniformed SMBC. However, for other payloads, the payload estimation and tracking performance degraded but was never worse than an uninformed SMBC. The degradation in payload estimates can be solved by using a finer payload discretization. The initial experimental evaluations validated the simulation results, after a slight modification and retuning . Overall, the OL/MMBC has demonstrated the ability to adapt quickly to uncertain payload information and maintain adequate tracking performance.

VI. Conclusions and Recommendations

6.1 Conclusions

Multiple Model Adaptive Estimation (MMAE) techniques have been successfully applied to adaptive model-based robot control. Combining the MMAE techniques with a model-based controller resulted in an adaptive model-based controller with the ability to adapt quickly to uncertain payload parameters and provide good tracking performance. This form of adaptive model-based control is known as open-loop Multiple Model-Based Control (OL/MMBC).

The OL/MMBC algorithm was simulated and experimentally evaluated for the first three links of the PUMA-560 manipulator. The initial evaluations of the OL/MMBC demonstrated that the algorithm quickly adapted to uncertain payloads resulting in tracking performance significantly better than a single model-based control (SMBC) not informed of the payload, and comparable to an artificially informed SMBC. Additionally, simulation and experimental results showed that, for certain payloads, the MMAE had difficulty in obtaining an accurate estimate. This was a result of using too coarse a payload parameter discretization.

During the experimental evaluations, tests were conducted using two sets of PD gains to determine what effects they would have on the performance of the OL/MMBC. The experimental results showed that, for both sets of gains, the MMAE quickly estimated the uncertain payload. Additionally, the tracking performance was comparable to an informed SMBC for both sets of gains. This showed that the choice of PD gains had no effect on the performance of the OL/MMBC.

An initial goal of this research was to evaluate experimentally the original closed-loop MMBC algorithm which used a closed-loop form of MMAE (ΔMMAE) that estimates parameter variations. Further analysis of the closed-loop MMBC algorithm revealed that applications are limited to robot manipulators whose tracking performance and dynamics depend heavily on the payload. This not the case for the PUMA-560 manipulator which is representative of the type of manipulator to be used for future Air Force applications.

Therefore, the original closed-loop MMBC is not suitable for the types of applications the Air Force is interested in.

An objective of the AFIT Robotics Systems Laboratory is develop and evaluate various forms of control algorithms with the capabilities to adapt to uncertainties in the environment and still provide accurate high speed tracking performance. These capabilities will make it possible for manipulators to be used in future Air Force applications such as robot telepresence. The OL/MMBC algorithm has demonstrated the potential to meet that objective.

6.2 Recommendations

Regarding the limited applications of the closed-loop MMBC, an area of further research would be to evaluate the algorithm on a manipulator whose tracking performance and dynamics depend heavily on the payload.

The initial evaluations of OL/MMBC algorithm opened the area for possible further research. For certain payloads, the MMAE had difficulty providing a good payload estimate. This was assumed to be a result of the payload discretization levels used in developing the filters. Therefore, one area of further research would be to look into an optimum selection of payload parameter discretization. An appropriate discretized payload parameter space should improve payload estimation. Recent research conducted by Sheldon developed a means of optimally discretizing the parameter space for MMAE and Multiple Model Adaptive Control applications [25].

During the OL/MMBC evaluation, the plant matrix, state transition matrix, covariance matrices, and Kalman filter gains were precomputed. This allowed for the experimental evaluations at a sample period of 9.9 ms. The use of parallel processing or application specific integrated circuits should allow for the the entire algorithm to be computed on-line in real-time. However, this does not preclude using good engineering judgment to reduce the complexity of the model used by the Kalman filters in order to achieve the same performance. Therefore an investigation into possible filter state reduction or reduction in model complexity should be investigated.

Finally, the MMAE techniques used for adaptive model-based robot control should be compared to other adaptive techniques proposed in the literature. This can be accomplished due to the experimental environment available in the Air Force Institute of Technology Robotic Systems Laboratory.

Appendix A. *Single Model-Based Control Plots*

The plots generated for simulations of a single model-based control (SMBC) algorithm are contained in this appendix. These plots were used as part of the analysis performed in Chapter 3, Section 3.4. These plots include the two trajectory profiles, torque comparision plots, and position error profiles.

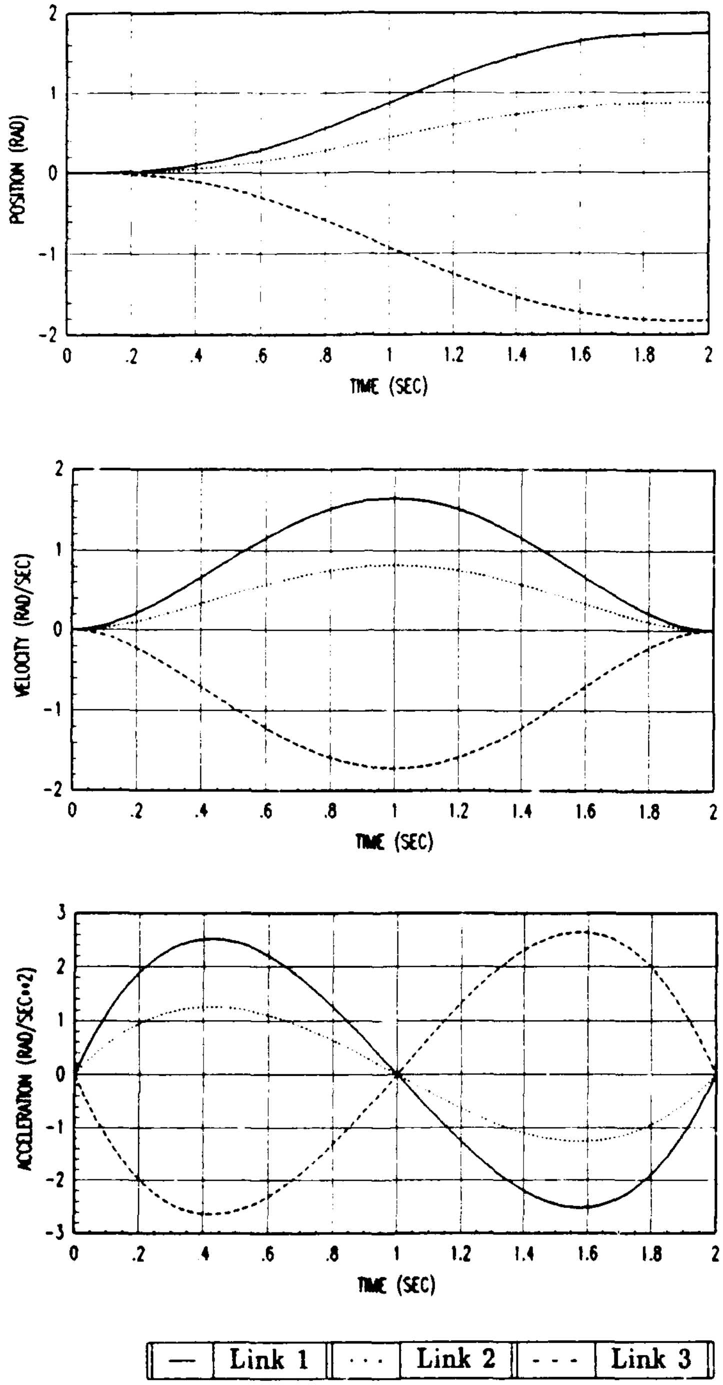


Figure A.1. Trajectory One: Position, Velocity, and Acceleration Profiles

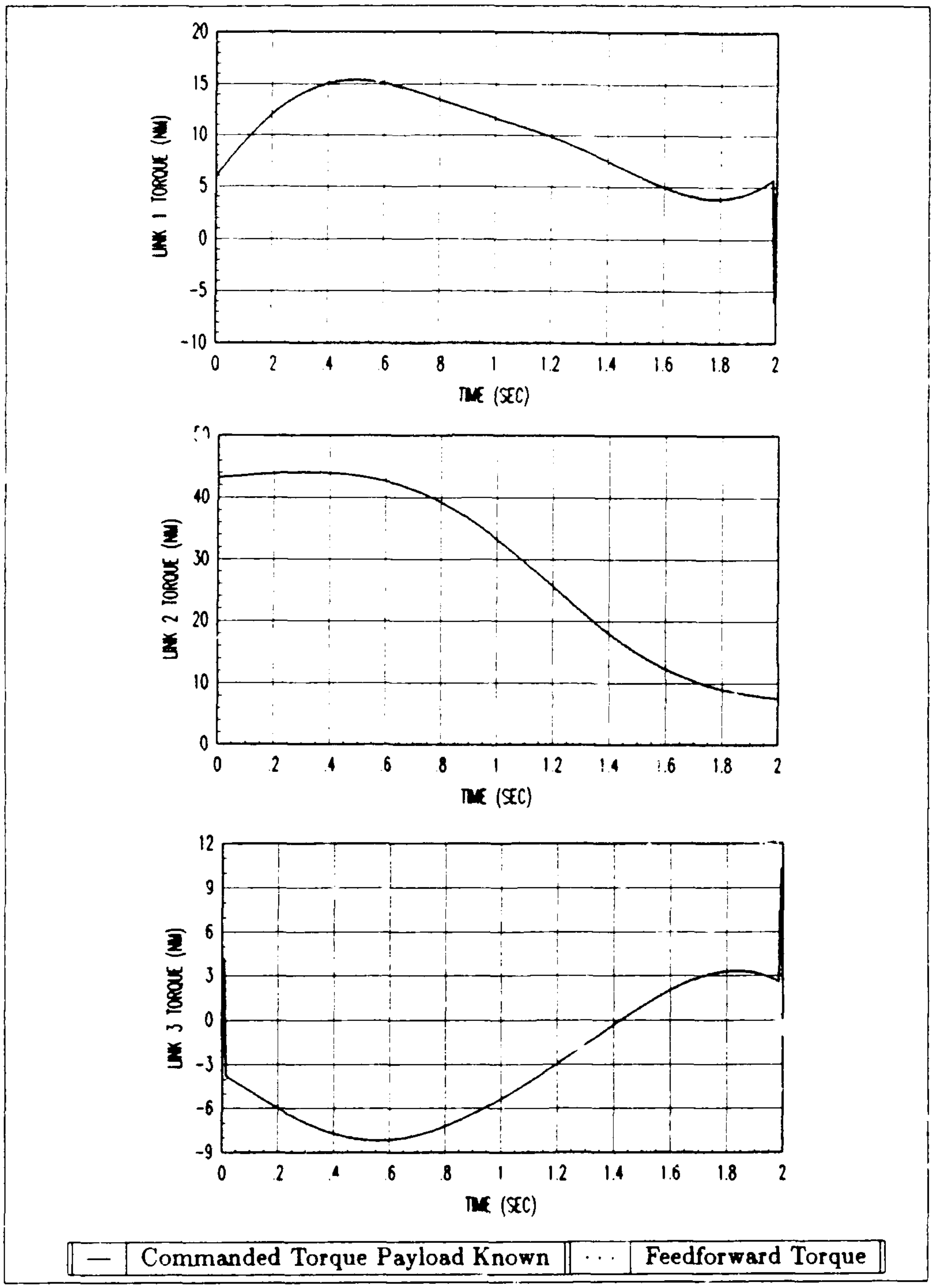


Figure A.2. Trajectory One: Torque Comparision External Payload of 0.0 Kg

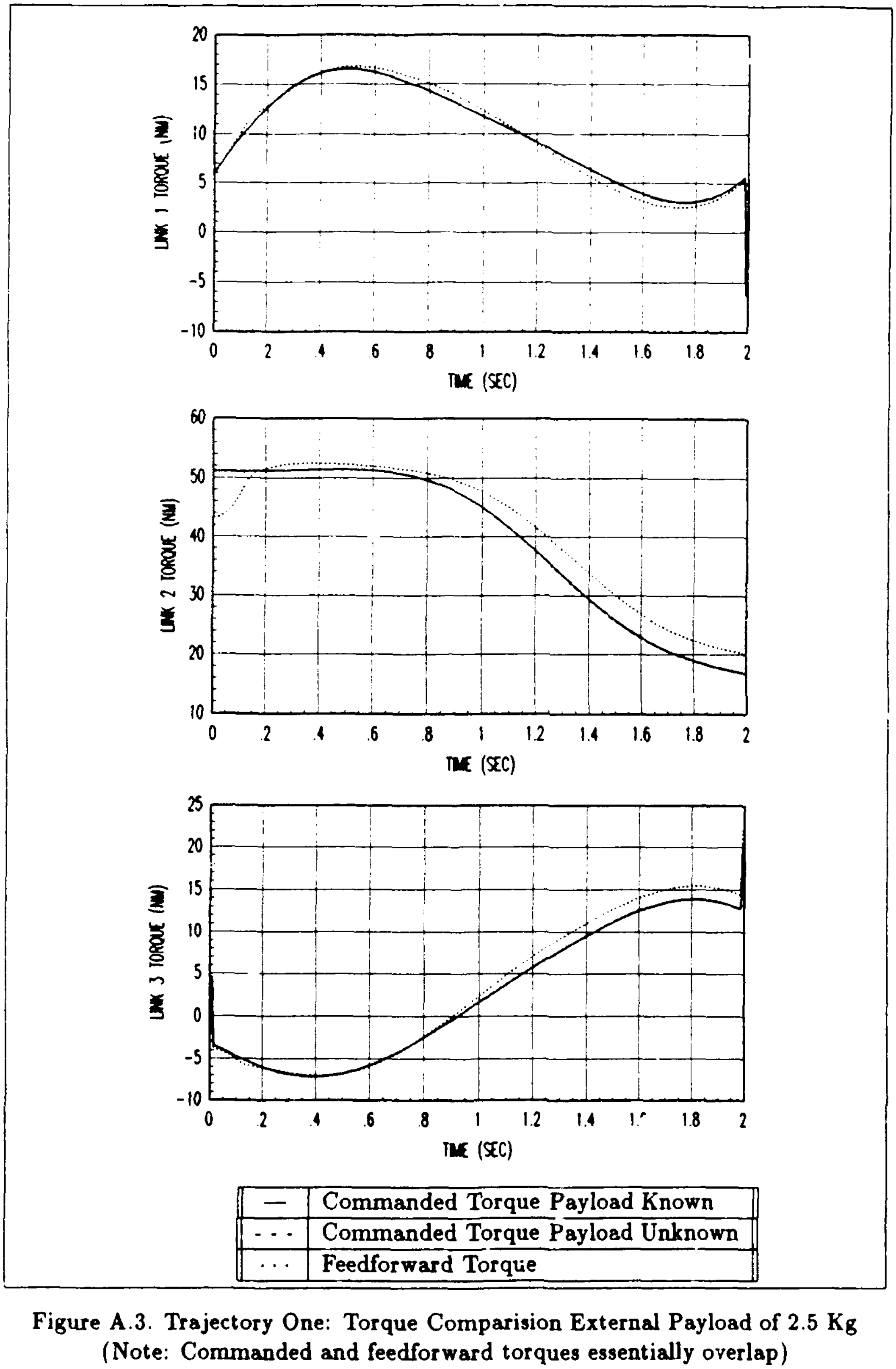


Figure A.3. Trajectory One: Torque Comparision External Payload of 2.5 Kg (Note: Commanded and feedforward torques essentially overlap)

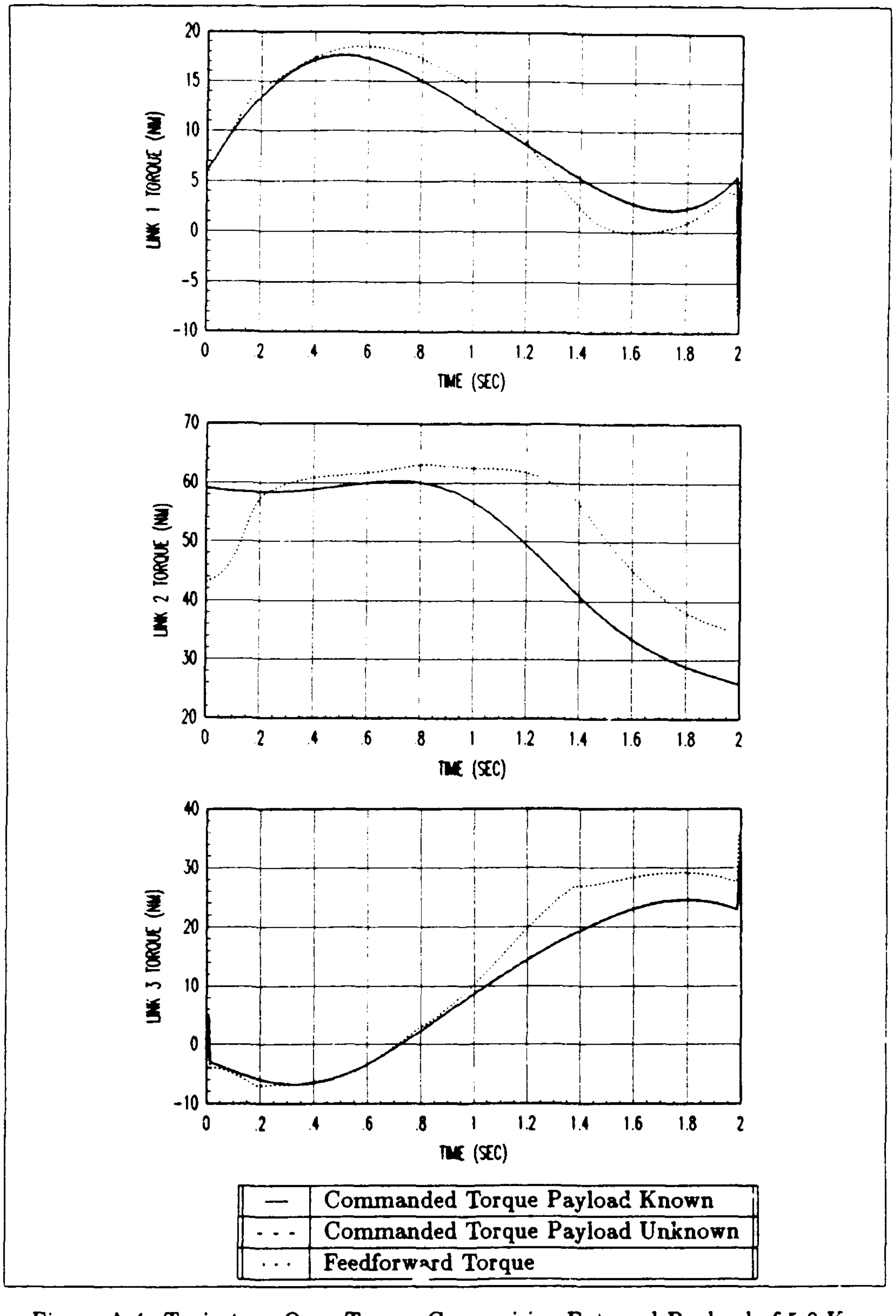


Figure A.4. Trajectory One: Torque Comparision External Payload of 5.0 Kg (Note: Commanded and feedforward torques essentially overlap)

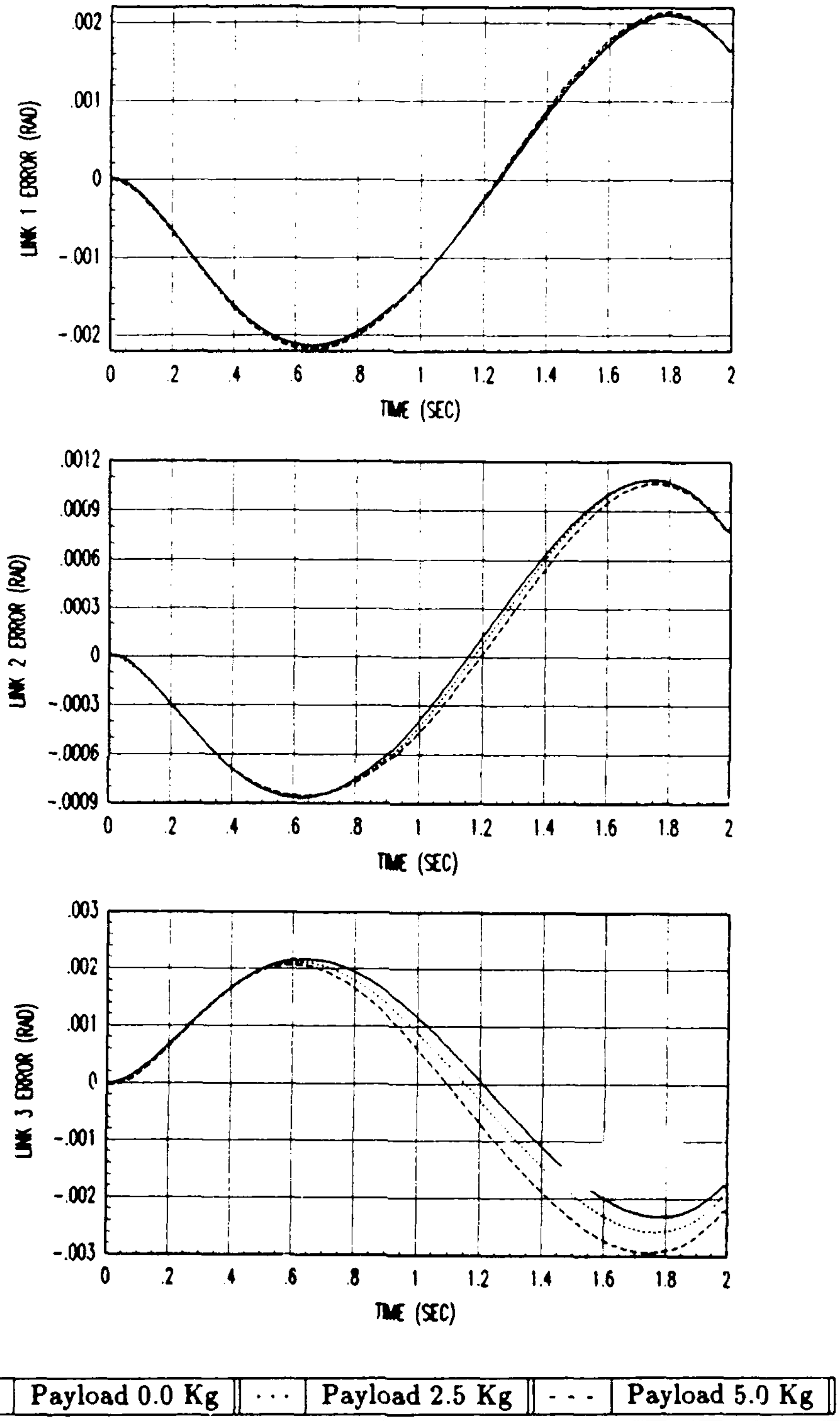


Figure A.5. Trajectory One: Position Error Profiles Payload Known

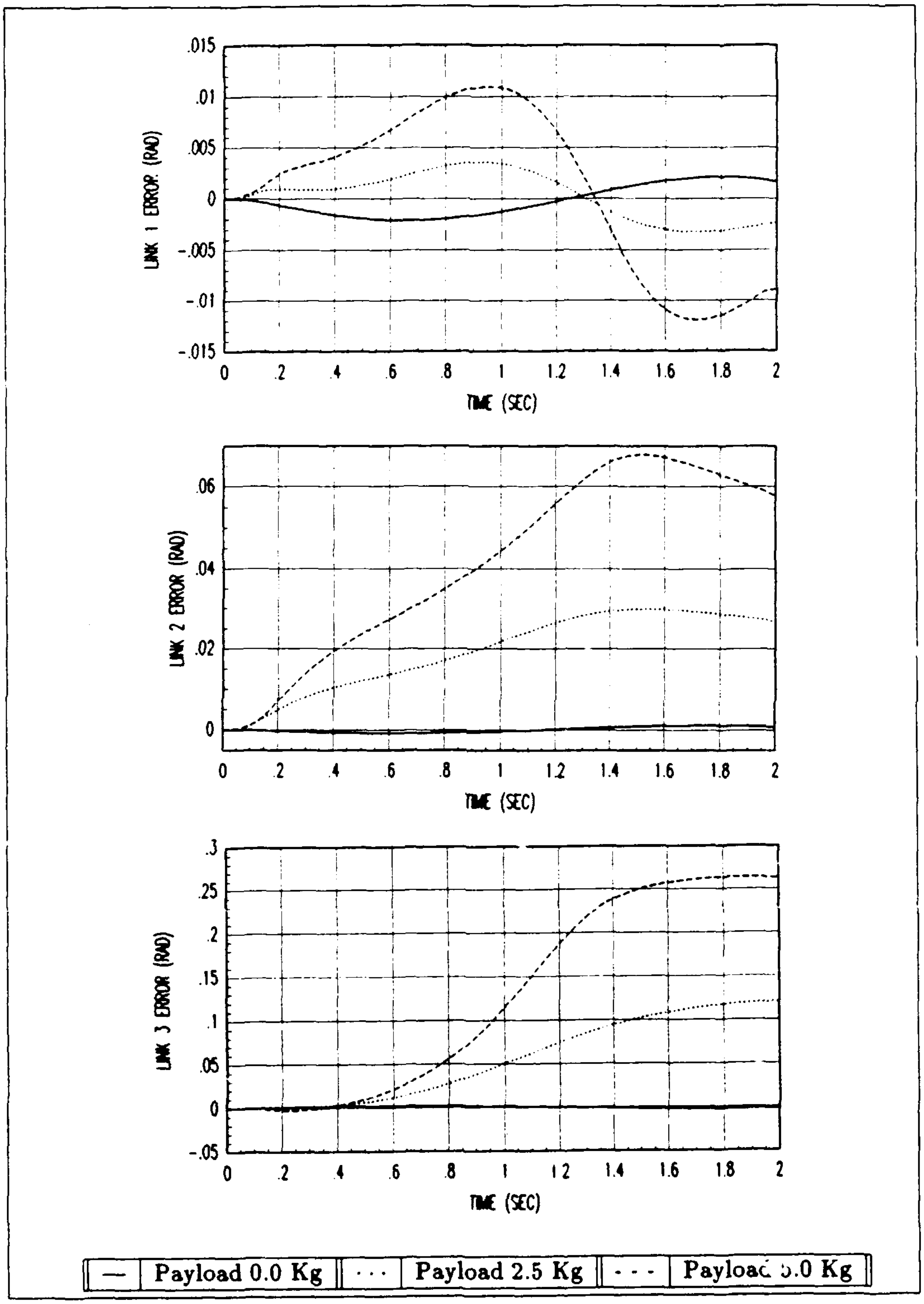


Figure A.6. Trajectory One: Position Error Profiles Payload Unknown

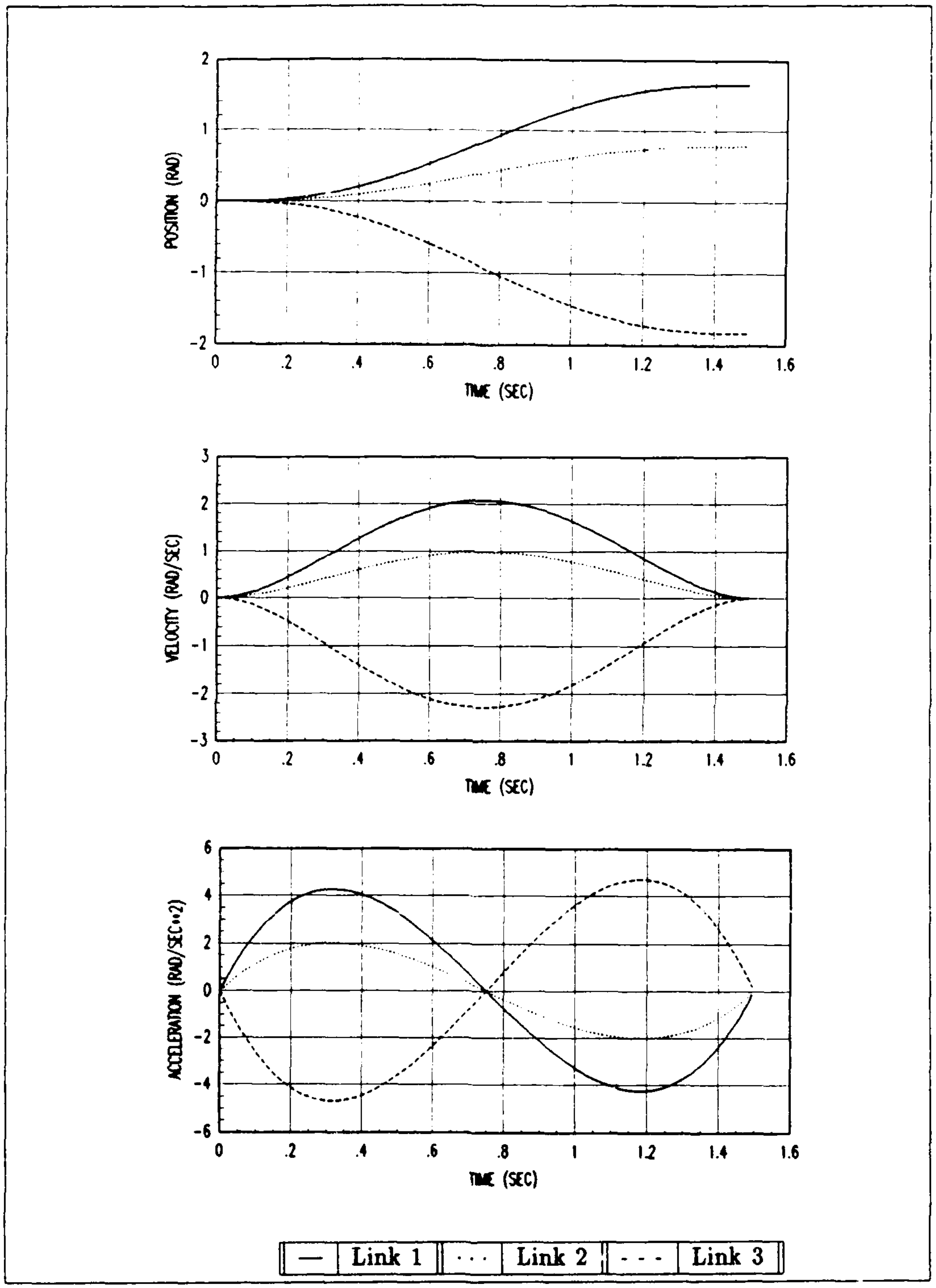


Figure A.7. Trajectory Two: Position, Velocity, and Acceleration Profile

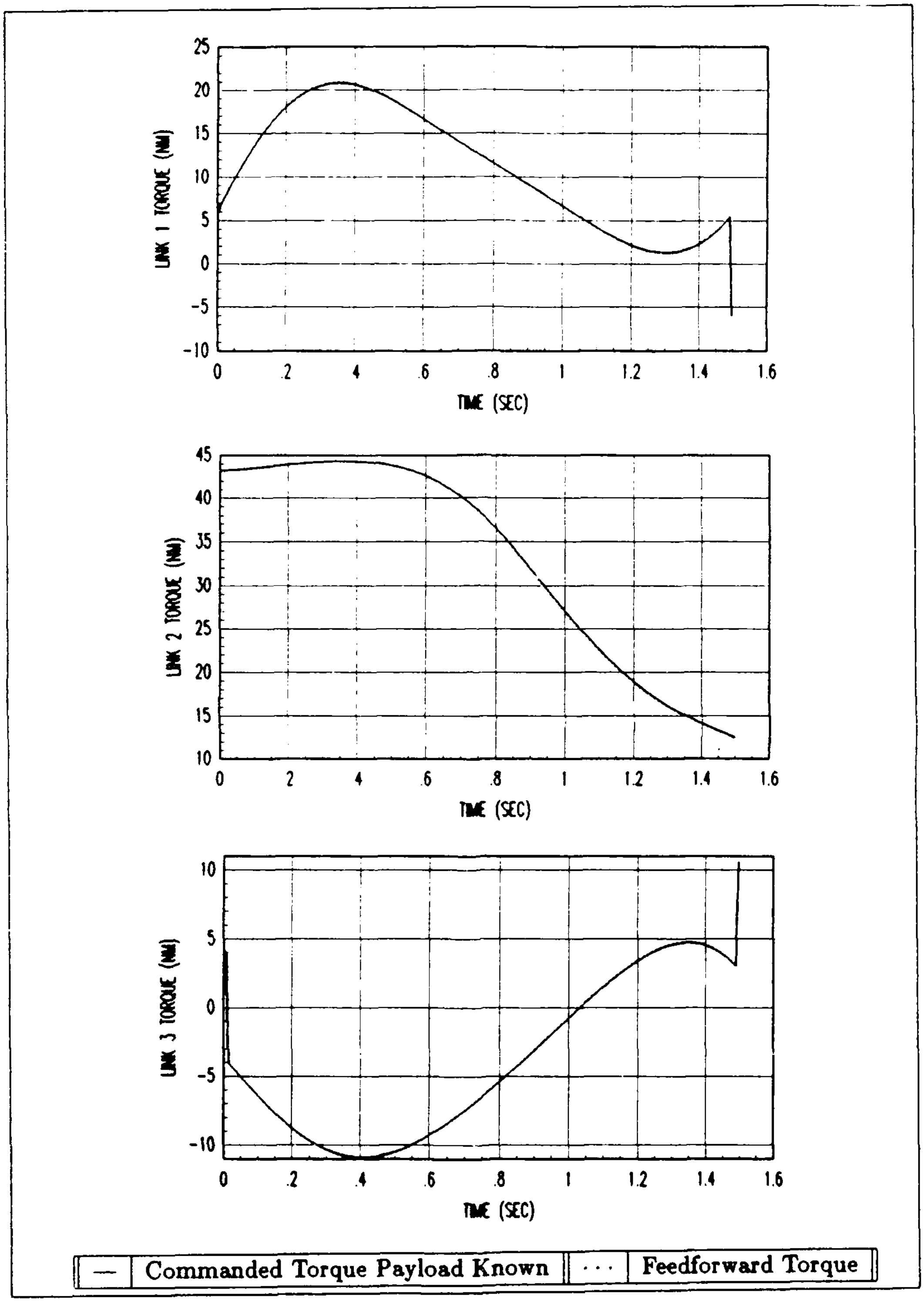


Figure A.8. Trajectory Two: Torque Comparision External Payload of 0.0 Kg (Note: Commanded and feedforward torques essentially overlap)

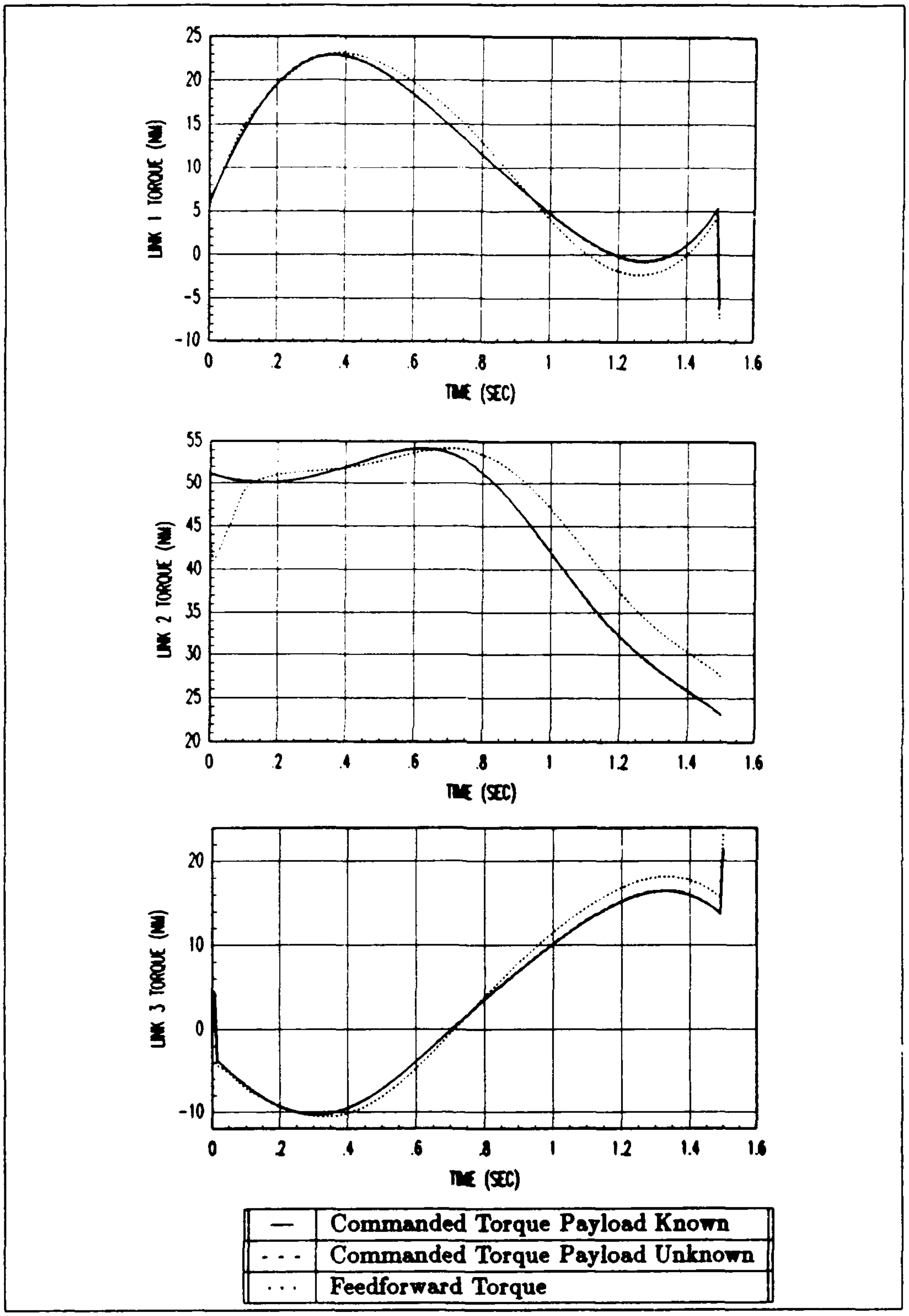


Figure A.9. Trajectory Two: Torque Comparision External Payload of 2.5 Kg (Note: Commanded and feedforward torques essentially overlap)

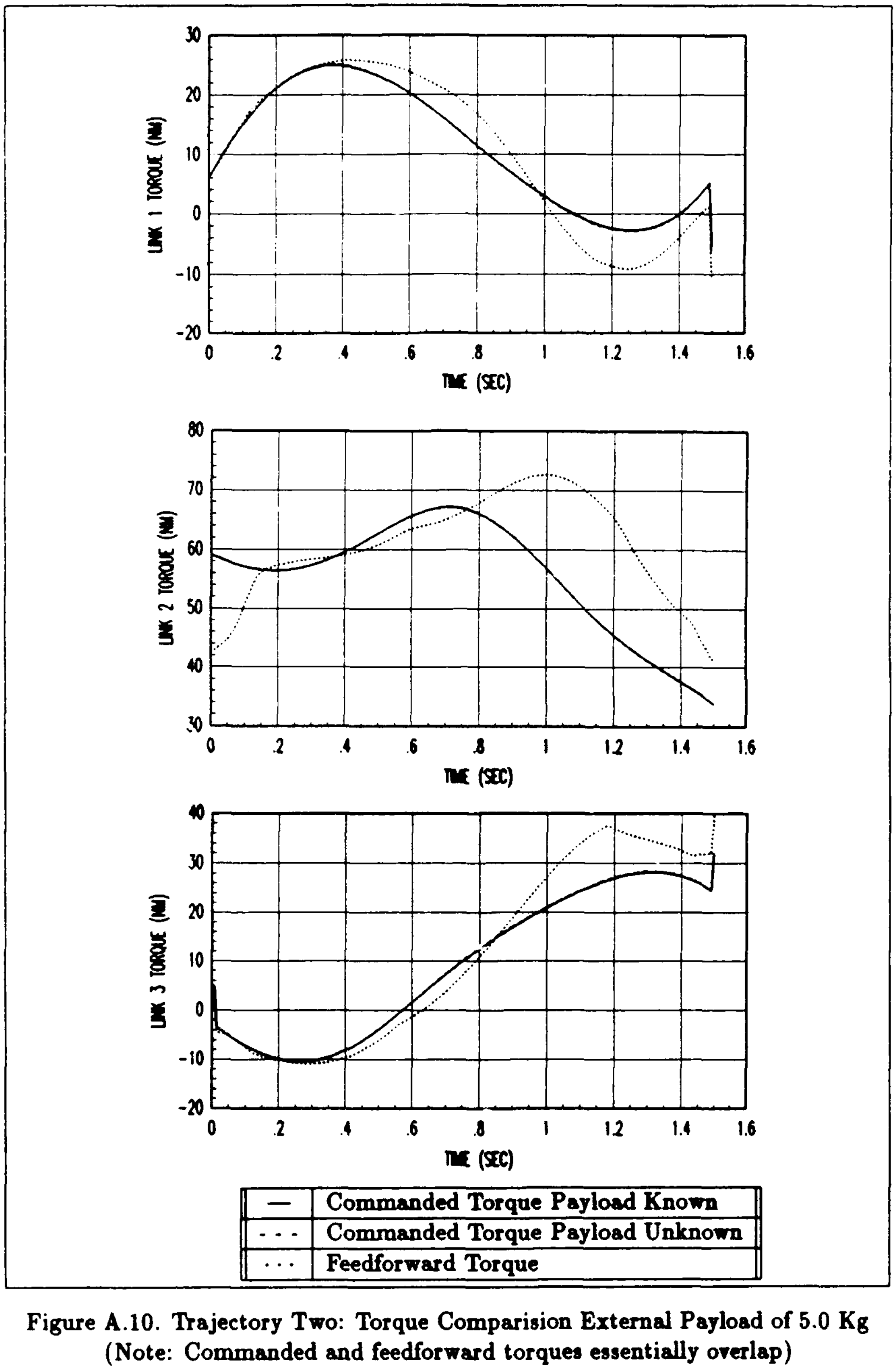


Figure A.10. Trajectory Two: Torque Comparision External Payload of 5.0 Kg (Note: Commanded and feedforward torques essentially overlap)

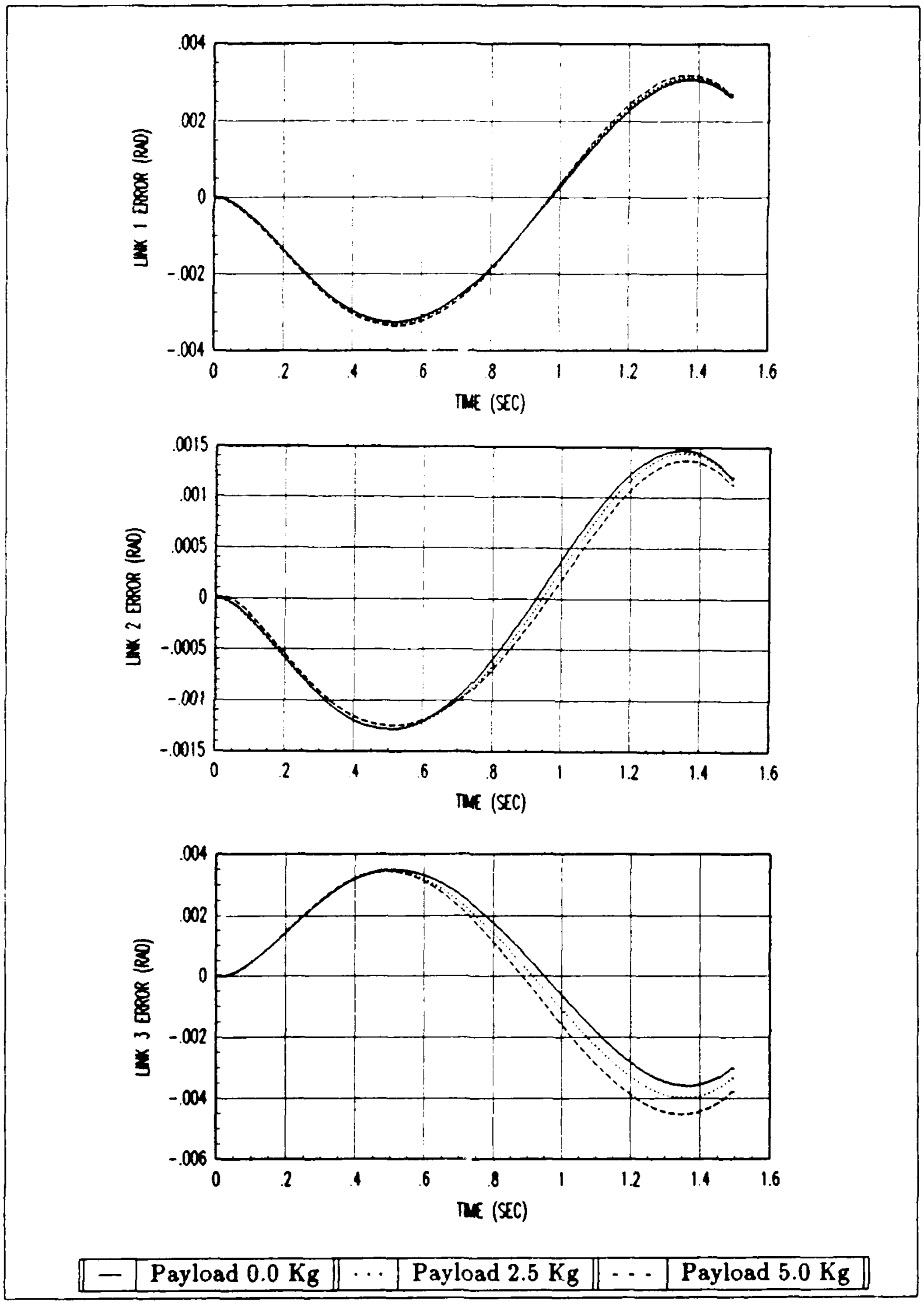


Figure A.11. Trajectory Two: Position Error Profiles Payload Known

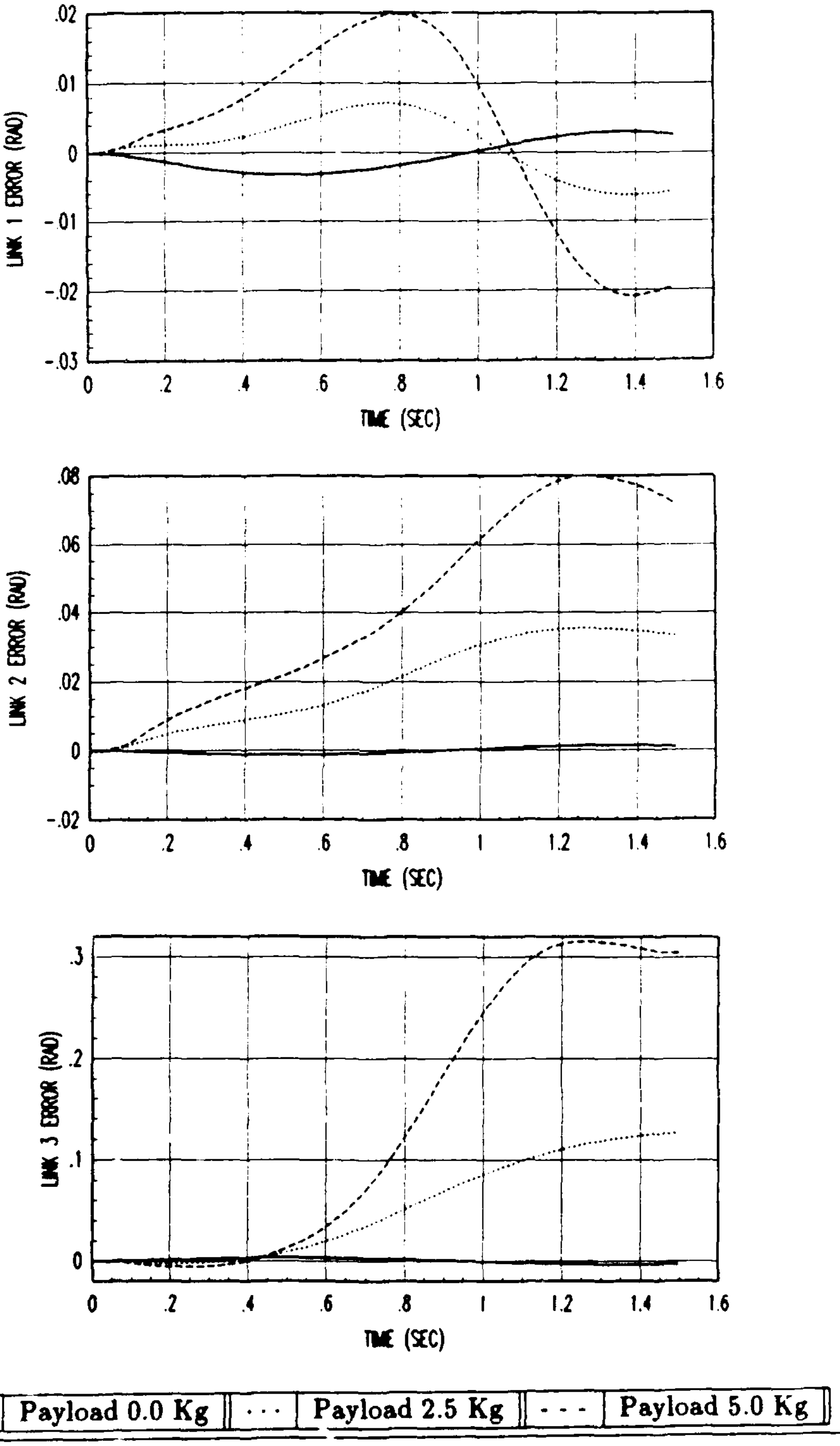


Figure A.12. Trajectory Two: Position Error Profiles Payload Unknown

Appendix B. *Details of The Linearized Plant Matrix*

This appendix develops the details of the lower partition of the linearized plant matrix $\mathbf{F}'$ shown in Equation (5.1). The lower partition of the linearized plant matrix was written as $\frac{\partial \mathbf{f}}{\partial \mathbf{x}}$ where $\mathbf{x}$ is the joint positions and velocities state vector. Remember from Equation (3.2) that $\mathbf{f}$ can be written as:

$$\mathbf{f} = \left[\mathbf{D}(\mathbf{q}, \mathbf{a}, t) + \mathbf{N}^2\mathbf{J}_m\right]^{-1}\left[\mathbf{N}\boldsymbol{\tau}(t) - \mathbf{h}(\dot{\mathbf{q}}, \mathbf{q}, \mathbf{a}) - \mathbf{N}^2\boldsymbol{\beta}_{\mathbf{m}}\dot{\mathbf{q}} - \mathbf{g}(\mathbf{q}, \mathbf{a}) - \boldsymbol{\tau}_s\right] \tag{B.1}$$

Then taking the partial derivative of Equation (B.1) with respect to the three joint positions (q_1, q_2, q_3) and three joint velocities $(\dot{q}_1, \dot{q}_2, \dot{q}_3)$ results in the following equations:

$$\begin{aligned} \frac{\partial \mathbf{f}}{\partial q_1} &= \frac{\partial\left[\mathbf{D} + \mathbf{N}^2\mathbf{J}_m\right]^{-1}}{\partial q_1}\left[\mathbf{N}\boldsymbol{\tau} - \mathbf{h} - \mathbf{N}^2\boldsymbol{\beta}_{\mathbf{m}}\dot{\mathbf{q}} - \mathbf{g} - \boldsymbol{\tau}_s\right] \\ &\quad + \left[\mathbf{D} + \mathbf{N}^2\mathbf{J}_m\right]^{-1}\left[-\frac{\partial \mathbf{h}}{\partial q_1} - \frac{\partial \mathbf{g}}{\partial q_1}\right] \end{aligned} \tag{B.2}$$

$$\begin{aligned} \frac{\partial \mathbf{f}}{\partial q_2} &= \frac{\partial\left[\mathbf{D} + \mathbf{N}^2\mathbf{J}_m\right]^{-1}}{\partial q_2}\left[\mathbf{N}\boldsymbol{\tau} - \mathbf{h} - \mathbf{N}^2\boldsymbol{\beta}_{\mathbf{m}}\dot{\mathbf{q}} - \mathbf{g} - \boldsymbol{\tau}_s\right] \\ &\quad + \left[\mathbf{D} + \mathbf{N}^2\mathbf{J}_m\right]^{-1}\left[-\frac{\partial \mathbf{h}}{\partial q_2} - \frac{\partial \mathbf{g}}{\partial q_2}\right] \end{aligned} \tag{B.3}$$

$$\begin{aligned} \frac{\partial \mathbf{f}}{\partial q_3} &= \frac{\partial\left[\mathbf{D} + \mathbf{N}^2\mathbf{J}_m\right]^{-1}}{\partial q_3}\left[\mathbf{N}\boldsymbol{\tau} - \mathbf{h} - \mathbf{N}^2\boldsymbol{\beta}_{\mathbf{m}}\dot{\mathbf{q}} - \mathbf{g} - \boldsymbol{\tau}_s\right] \\ &\quad + \left[\mathbf{D} + \mathbf{N}^2\mathbf{J}_m\right]^{-1}\left[-\frac{\partial \mathbf{h}}{\partial q_3} - \frac{\partial \mathbf{g}}{\partial q_3}\right] \end{aligned} \tag{B.4}$$

$$\frac{\partial \mathbf{f}}{\partial \dot{q}_1} = \left[\mathbf{D} + \mathbf{N}^2\mathbf{J}_m\right]^{-1}\left[-\frac{\partial \mathbf{h}}{\partial \dot{q}_1} \cdot N_1^2\beta_{m_1}\right] \tag{B.5}$$

$$\frac{\partial \mathbf{f}}{\partial \dot{q}_2} = \left[\mathbf{D} + \mathbf{N}^2\mathbf{J}_m\right]^{-1}\left[-\frac{\partial \mathbf{h}}{\partial \dot{q}_2} - N_2^2\beta_{m_2}\right] \tag{B.6}$$

$$\frac{\partial \mathbf{f}}{\partial \dot{q}_3} = \left[\mathbf{D} + \mathbf{N}^2\mathbf{J}_m\right]^{-1}\left[-\frac{\partial \mathbf{h}}{\partial \dot{q}_3} - N_3^2\beta_{m_3}\right] \tag{B.7}$$

Appendix C. *OL/MMBC Simulation Plots*

This appendix contains the plots of simulations evaluations of the OL/MMBC algorithm on a PUMA-560 manipulator. The plots include in this appendix are the probability plots obtained during Kalman filter noise tuning and plots obtained to demonstrate payload estimation and tracking performance.

C.1 Filter Tuning Plots

The plots of the hypothesis conditional probabilities obtained from the tuning procedures described in Section 5.3.1 are contained in this section.

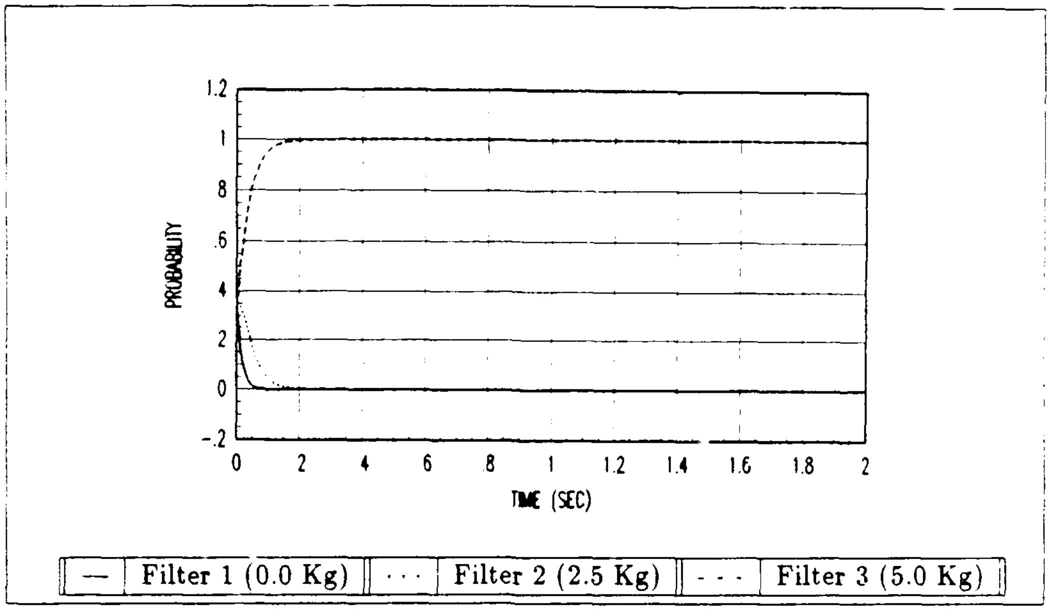


Figure C.1. Payload 5.0 Kg: Probability Profiles **Q** = 100, **R** = 10^{-6} and $p_k(t_{i-1})$ not reset to 1/3

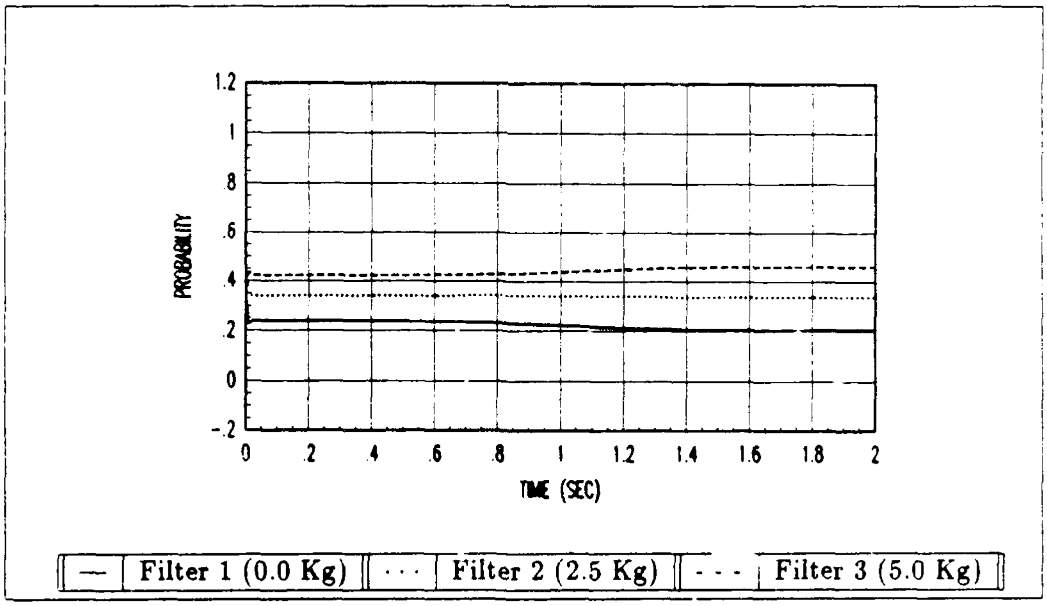


Figure C.2. Payload 5.0 Kg: Probability Profiles **Q** = 100, **R** = 10^{-6} and $p_k(t_{i-1})$ reset to 1/3

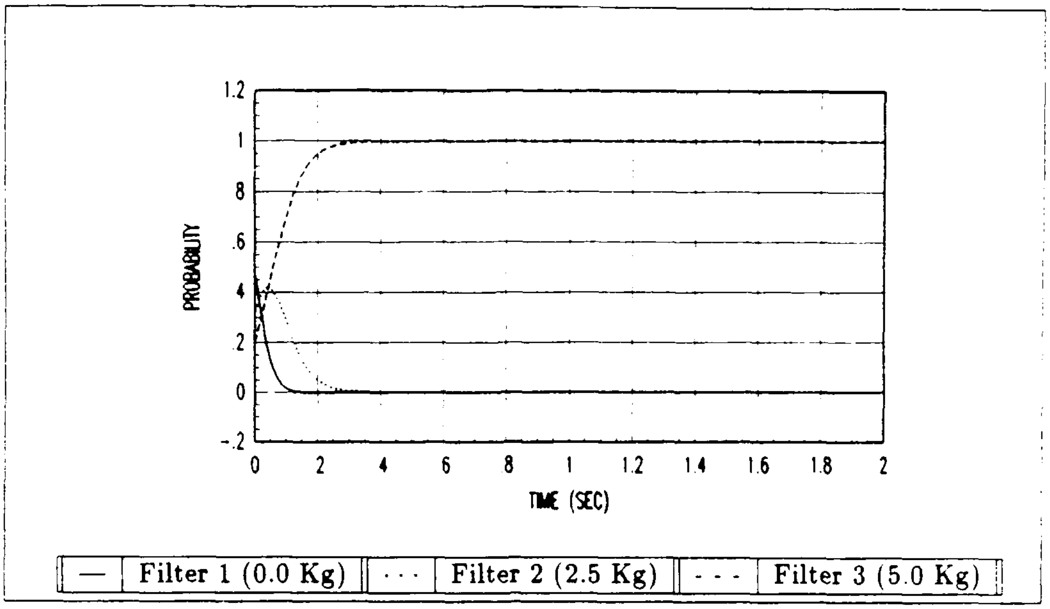


Figure C.3. Payload 5.0 Kg: Probability Profiles **Q** = 10, **R** = 10^{-6} and $p_k(t_{i-1})$ not reset to 1/3

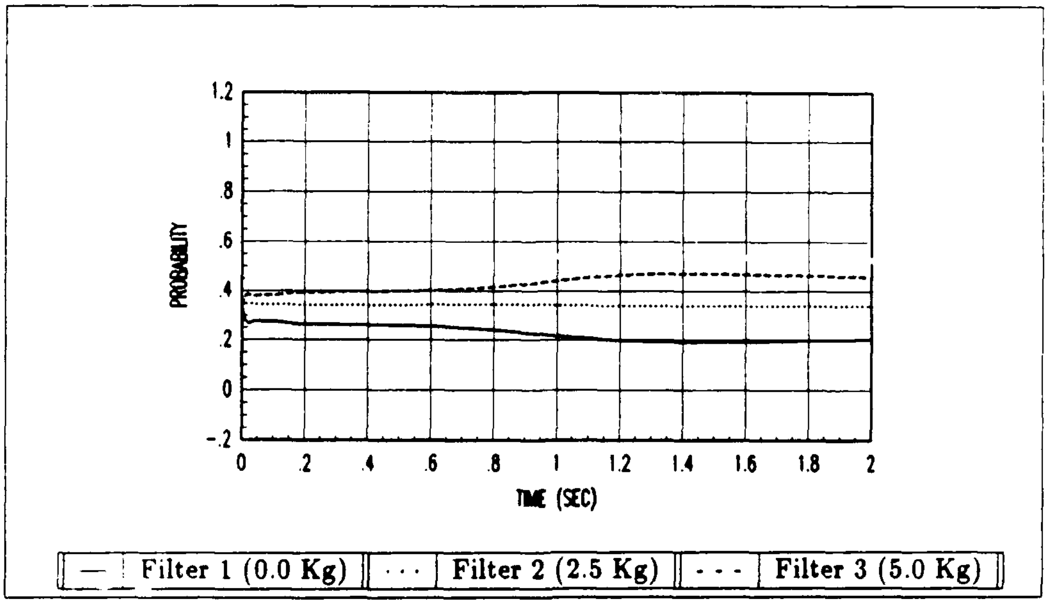


Figure C.4. Payload 5.0 Kg: Probability Profiles **Q** = 10, **R** = 10^{-6} and $p_k(t_{i-1})$ reset to 1/3

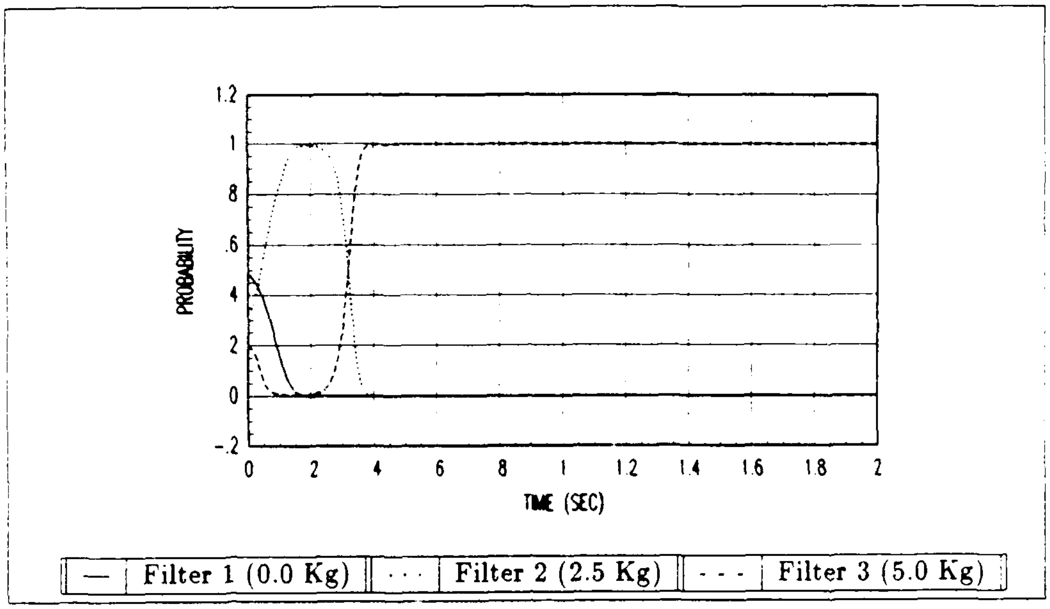


Figure C.5. Payload 5.0 Kg: Probability Profiles **Q** = 1, **R** = 10^{-6} and $p_k(t_{i-1})$ not reset to 1/3

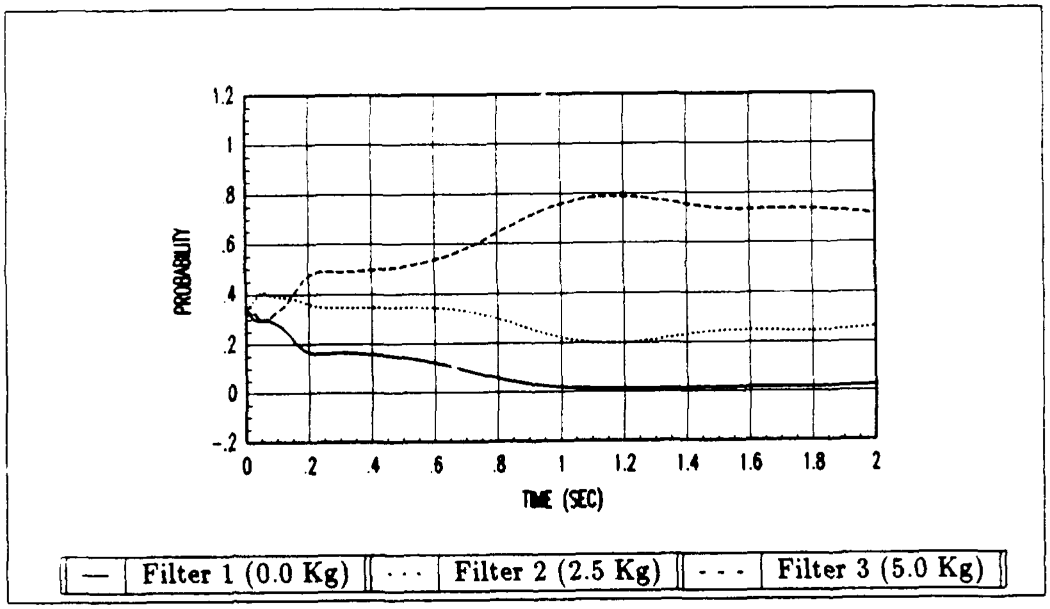


Figure C.6. Payload 5.0 Kg: Probability Profiles **Q** = 1, **R** = 10^{-6} and $p_k(t_{i-1})$ reset to 1/3

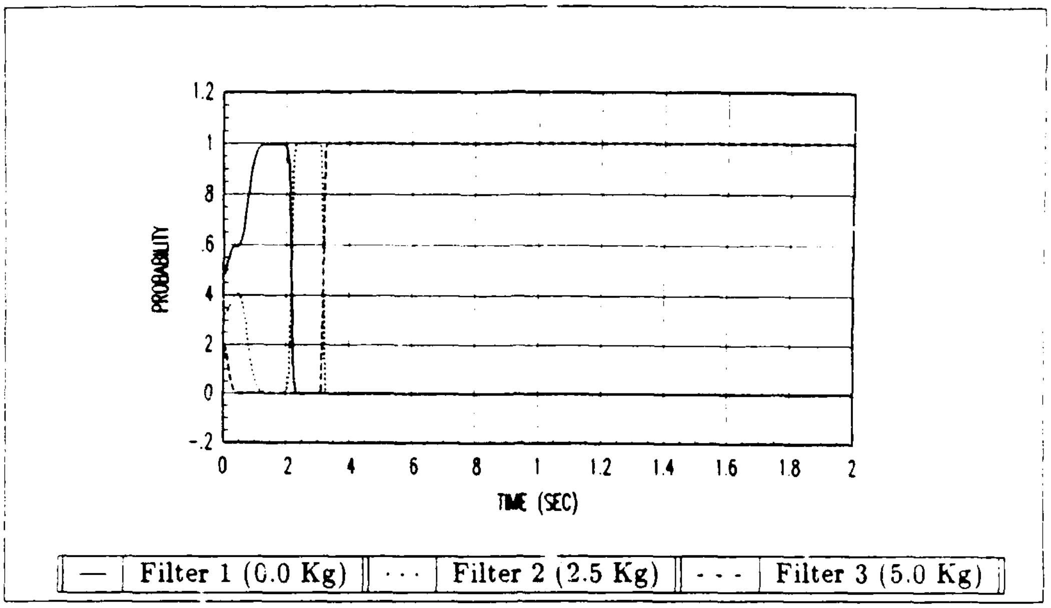


Figure C.7. Payload 5.0 Kg: Probability Profiles $\mathbf{Q} = 0.1$, $\mathbf{R} = 10^{-6}$ and $p_k(t_{i-1})$ not reset to 1/3

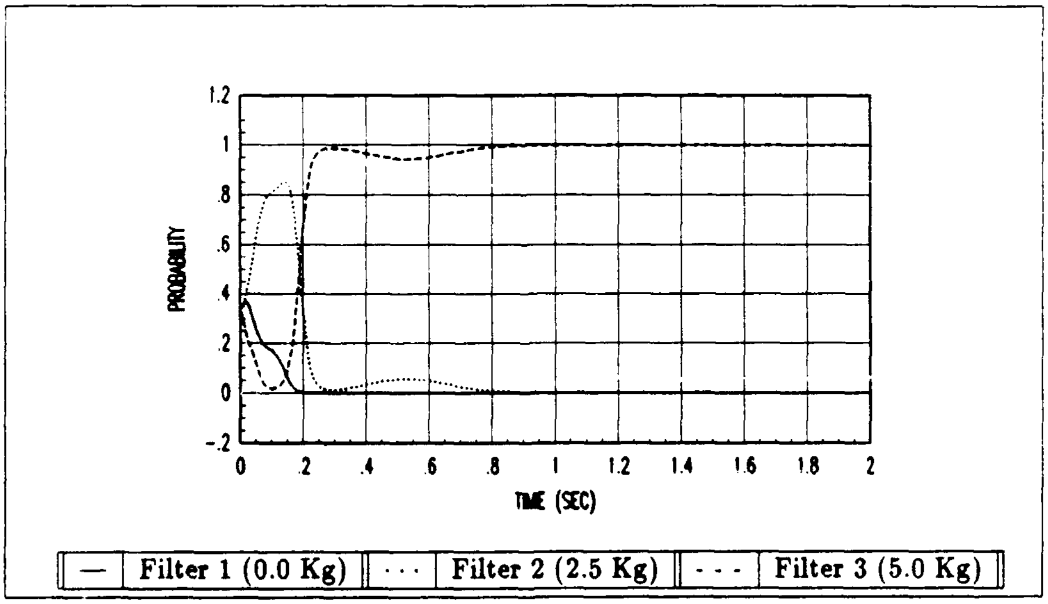


Figure C.8. Payload 5.0 Kg: Probability Profiles $\mathbf{Q} = 0.1$, $\mathbf{R} = 10^{-6}$ and $p_k(t_{i-1})$ reset to 1/3

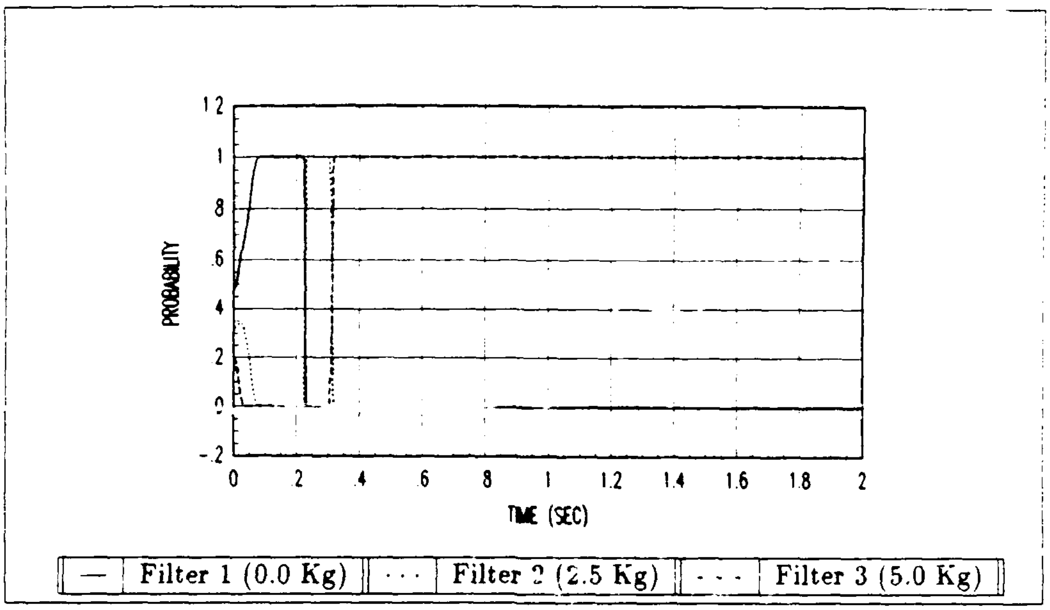


Figure C.9. Payload 5.0 Kg: Probability Profiles **Q** = 0.01, **R** = 10^{-6} and $p_k(t_{i-1})$ not reset to 1/3

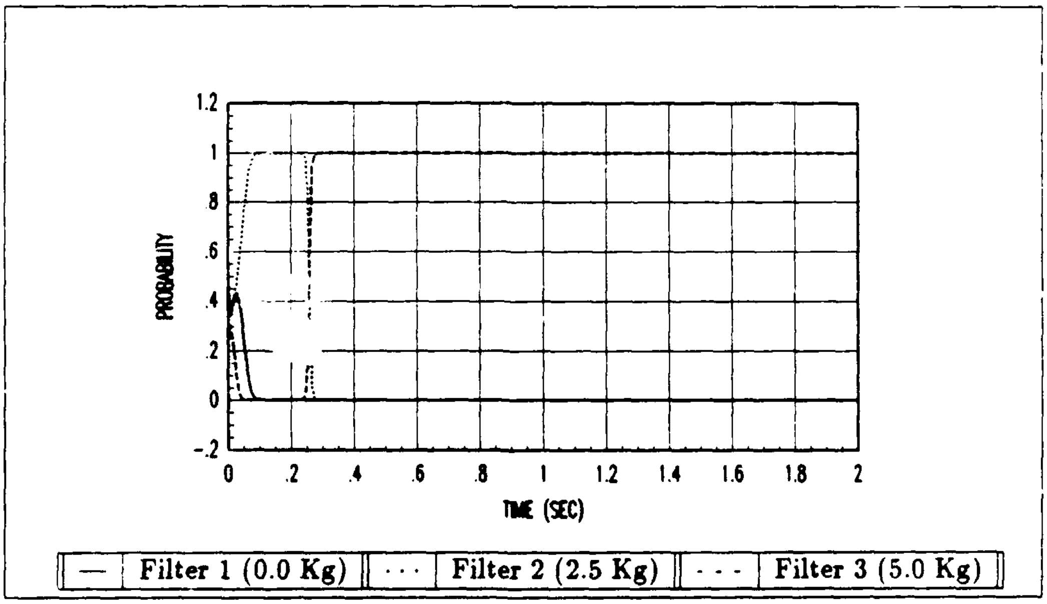


Figure C.10. Payload 5.0 Kg: Probability Profiles **Q** = 0.01, **R** = 10^{-6} and $p_k(t_{i-1})$ reset to 1/3

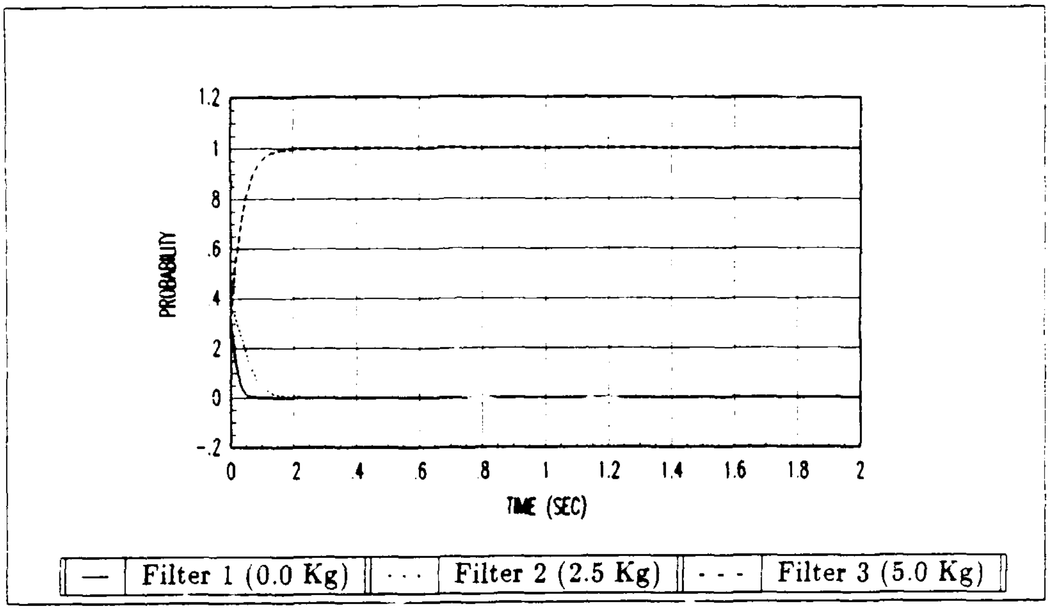


Figure C.11. Payload 2.5 Kg: Probability Profiles **Q** = 100, **R** = 10^{-6} and $p_k(t_{i-1})$ not reset to 1/3

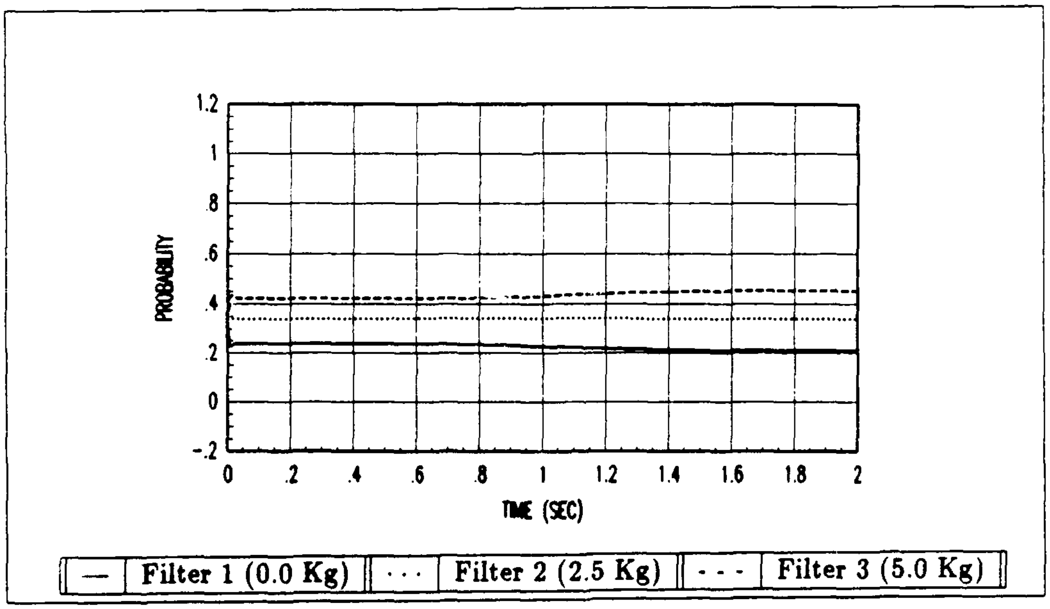


Figure C.12. Payload 2.5 Kg: Probability Profiles **Q** = 100, **R** = 10^{-6} and $p_k(t_{i-1})$ reset to 1/3

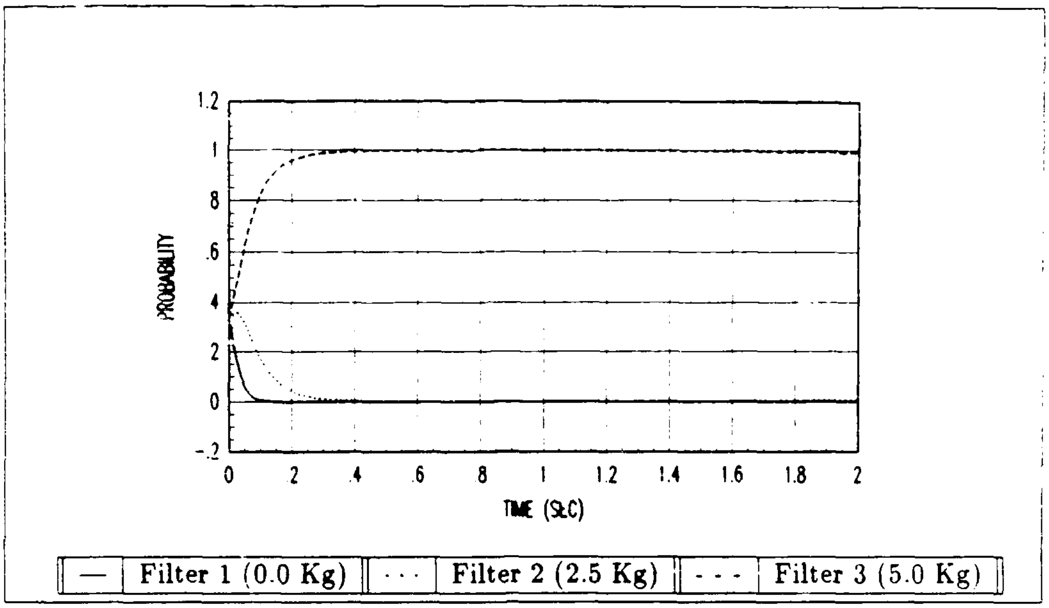


Figure C.13. Payload 2.5 Kg: Probability Profiles $\mathbf{Q} = 10$, $\mathbf{R} = 10^{-6}$ and $p_k(t_{i-1})$ not reset to 1/3

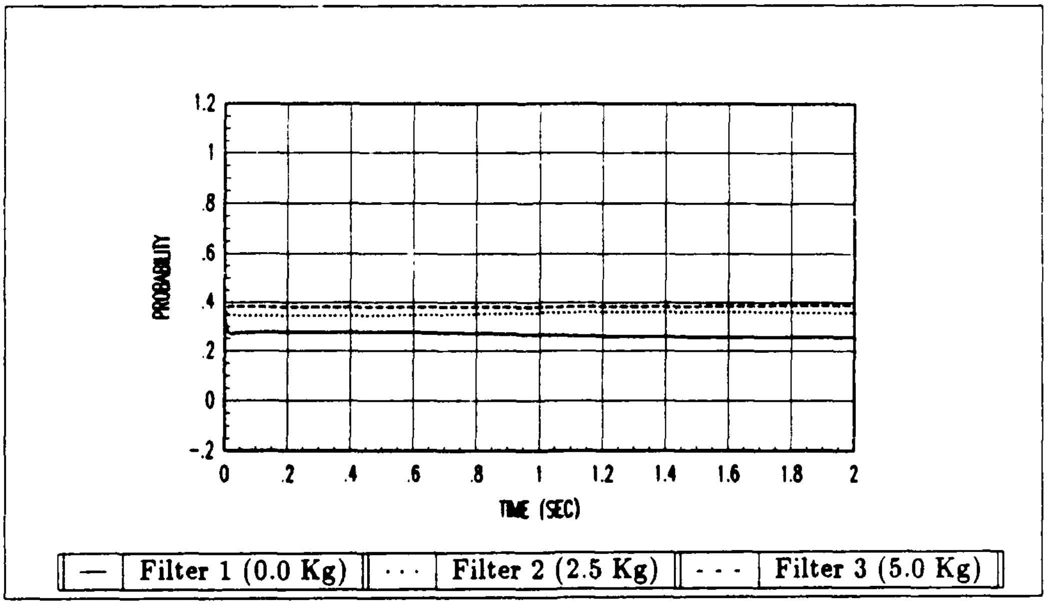


Figure C.14. Payload 2.5 Kg: Probability Profiles $\mathbf{Q} = 10$, $\mathbf{R} = 10^{-6}$ and $p_k(t_{i-1})$ reset to 1/3

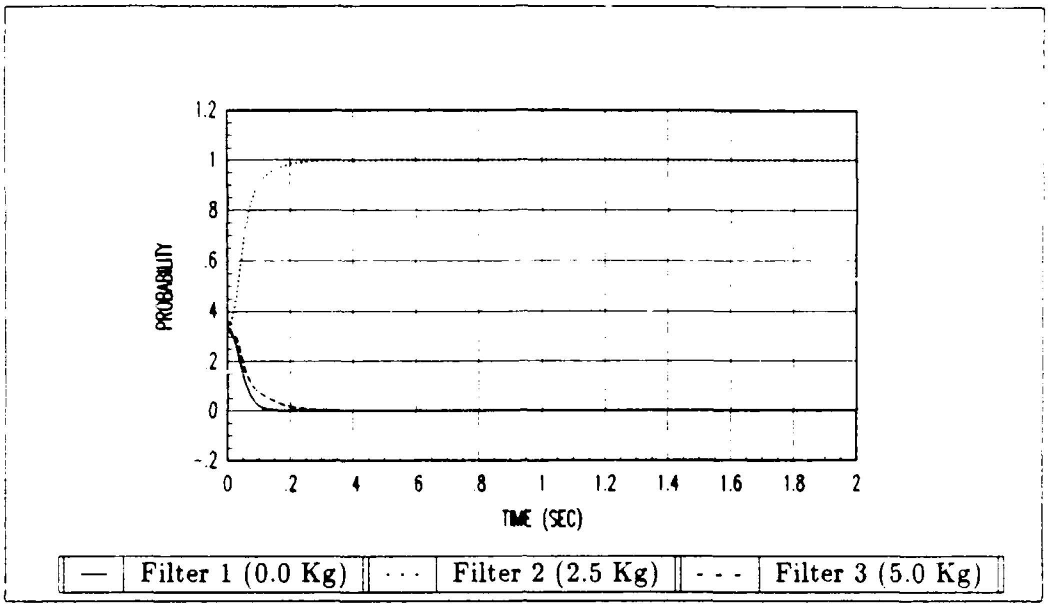


Figure C.15. Payload 2.5 Kg: Probability Profiles $\mathbf{Q} = 1$, $\mathbf{R} = 10^{-6}$ and $p_k(t_{i-1})$ not reset to 1/3

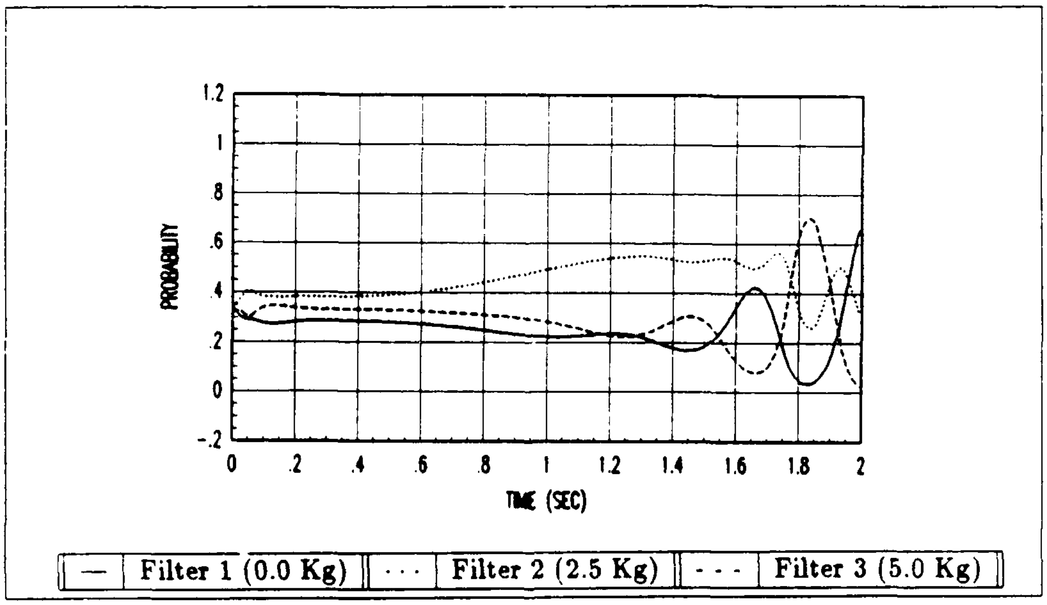


Figure C.16. Payload 2.5 Kg: Probability Profiles $\mathbf{Q} = 1$, $\mathbf{R} = 10^{-6}$ and $p_k(t_{i-1})$ reset to 1/3

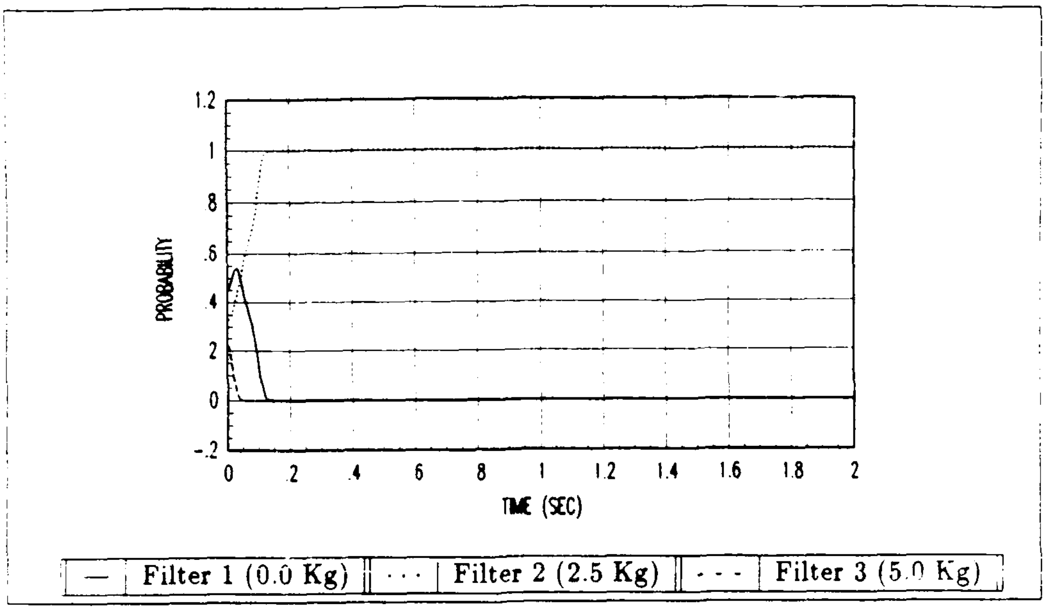


Figure C.17. Payload 2.5 Kg: Probability Profiles $\mathbf{Q} = 0.1$, $\mathbf{R} = 10^{-6}$ and $p_k(t_{i-1})$ not reset to 1/3

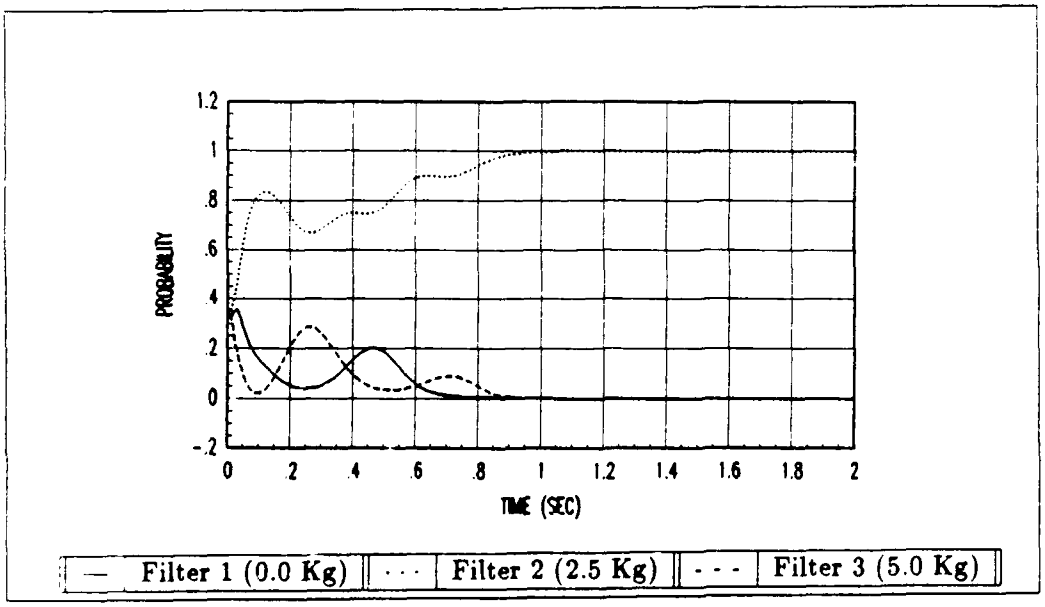


Figure C.18. Payload 2.5 Kg: Probability Profiles $\mathbf{Q} = 0.1$, $\mathbf{R} = 10^{-6}$ and $p_k(t_{i-1})$ reset to 1/3

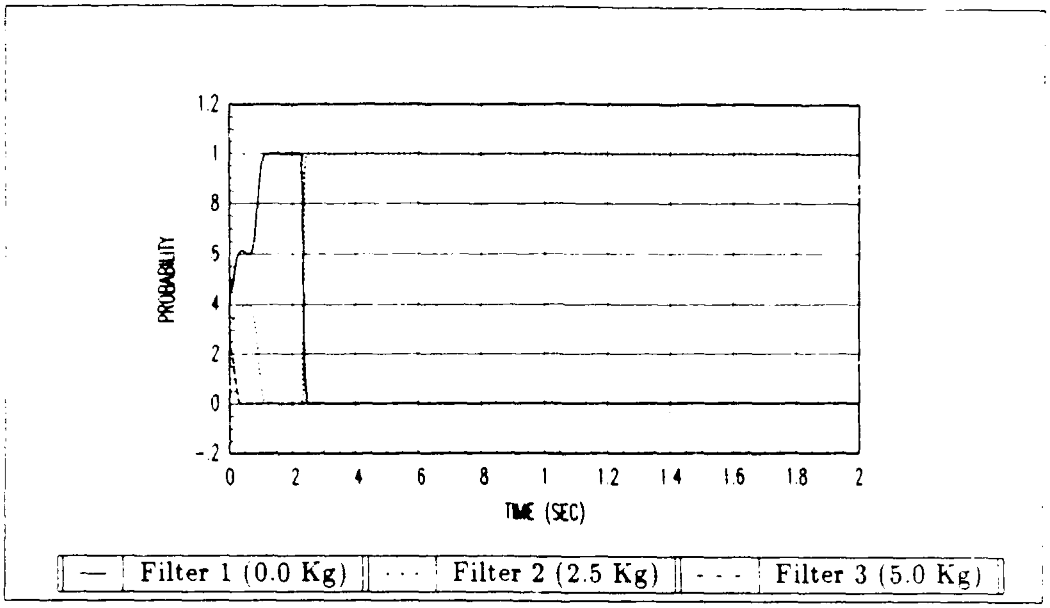


Figure C.19. Payload 2.5 Kg: Probability Profiles **Q** = 0.01, **R** = 10^{-6} and $p_k(t_{i-1})$ not reset to 1/3

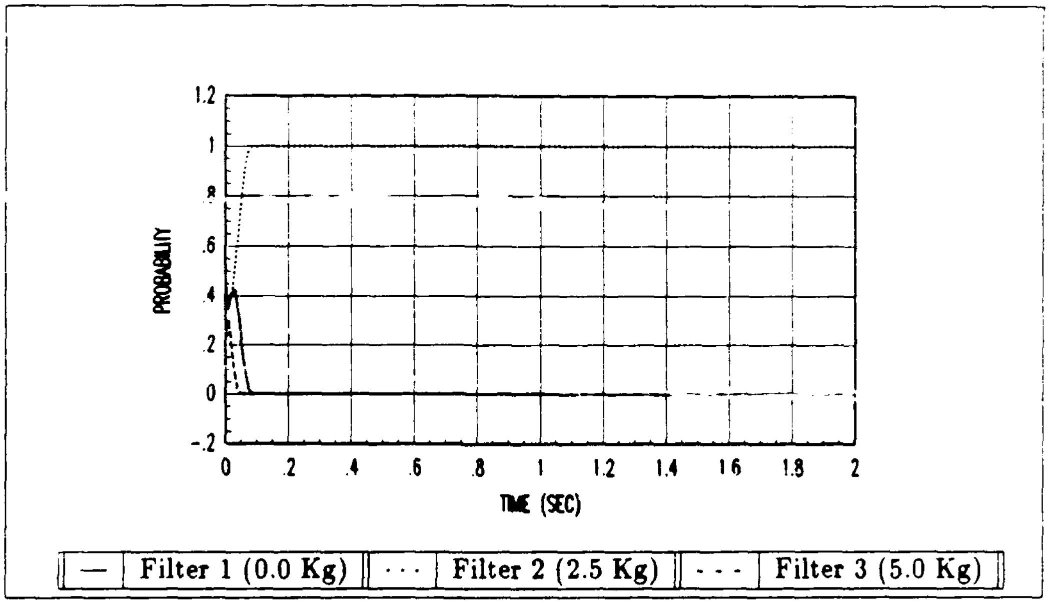


Figure C.20. Payload 2.5 Kg: Probability Profiles **Q** = 0.01, **R** = 10^{-6} and $p_k(t_{i-1})$ reset to 1/3

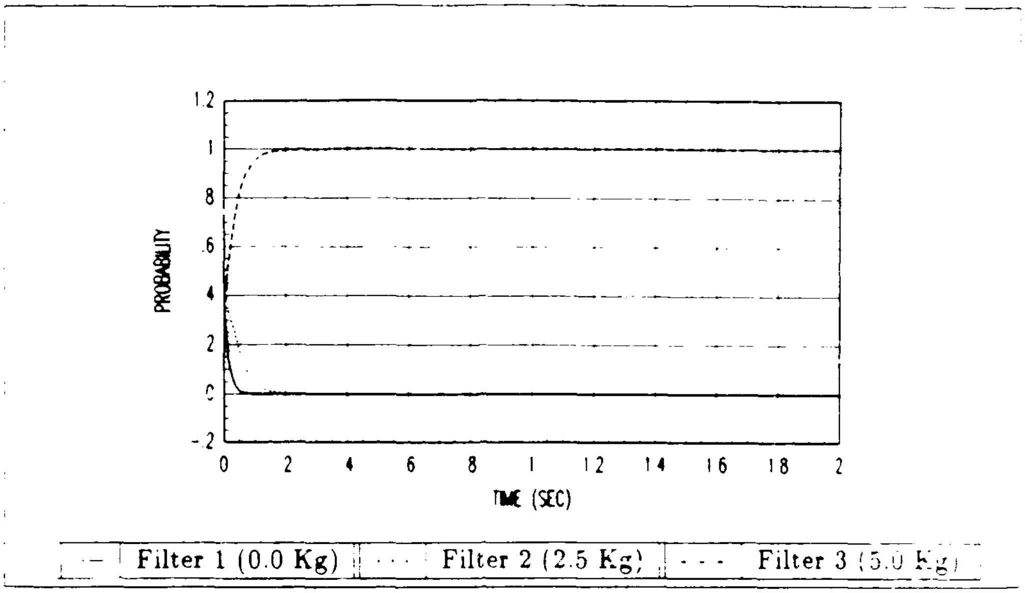


Figure C.21. Payload 0.0 Kg: Probability Profiles $\mathbf{Q} = 100$, $\mathbf{R} = 10^{-6}$ and $p_k(t_{i-1})$ not reset to 1/3

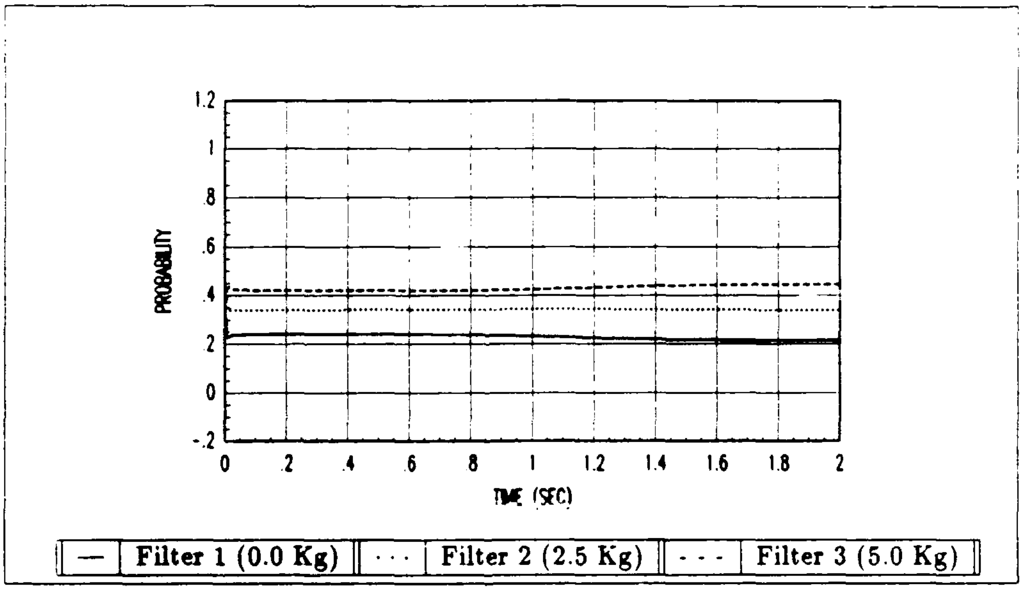


Figure C.22. Payload 0.0 Kg: Probability Profiles $\mathbf{Q} = 100$, $\mathbf{R} = 10^{-6}$ and $p_k(t_{i-1})$ reset to 1/3

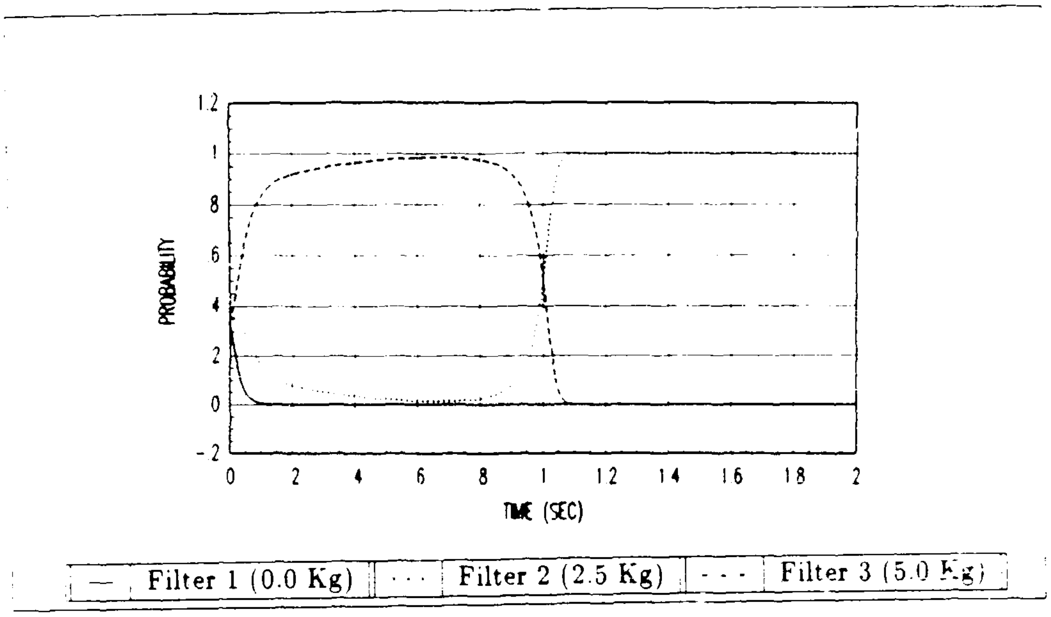


Figure C.23. Payload 0.0 Kg: Probability Profiles $\mathbf{Q} = 10$, $\mathbf{R} = 10^{-6}$ and $p_k(t_{i-1})$ not reset to 1/3

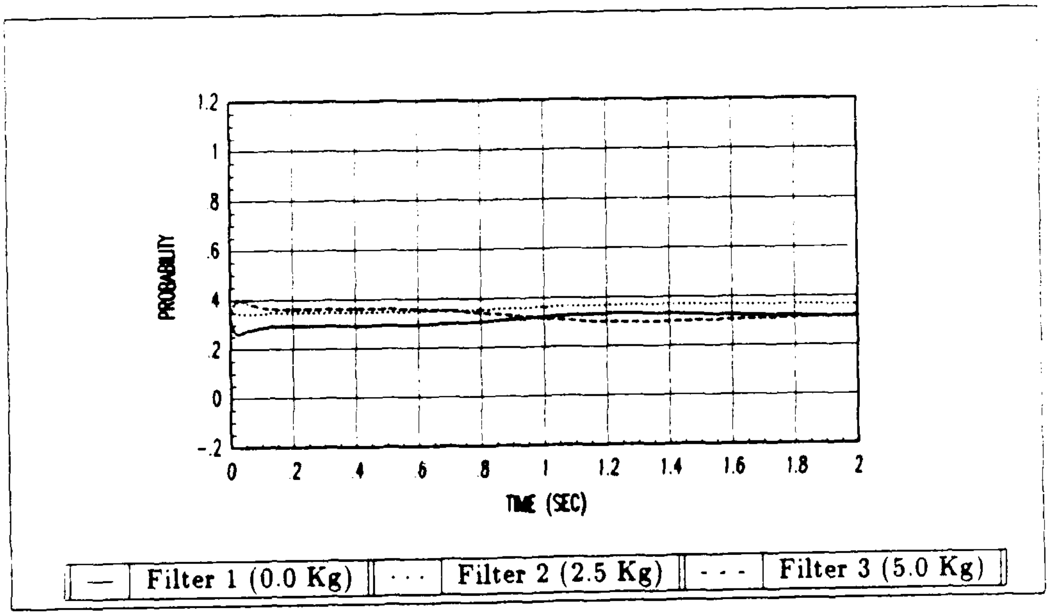


Figure C.24. Payload 0.0 Kg: Probability Profiles $\mathbf{Q} = 10$, $\mathbf{R} = 10^{-6}$ and $p_k(t_{i-1})$ reset to 1/3

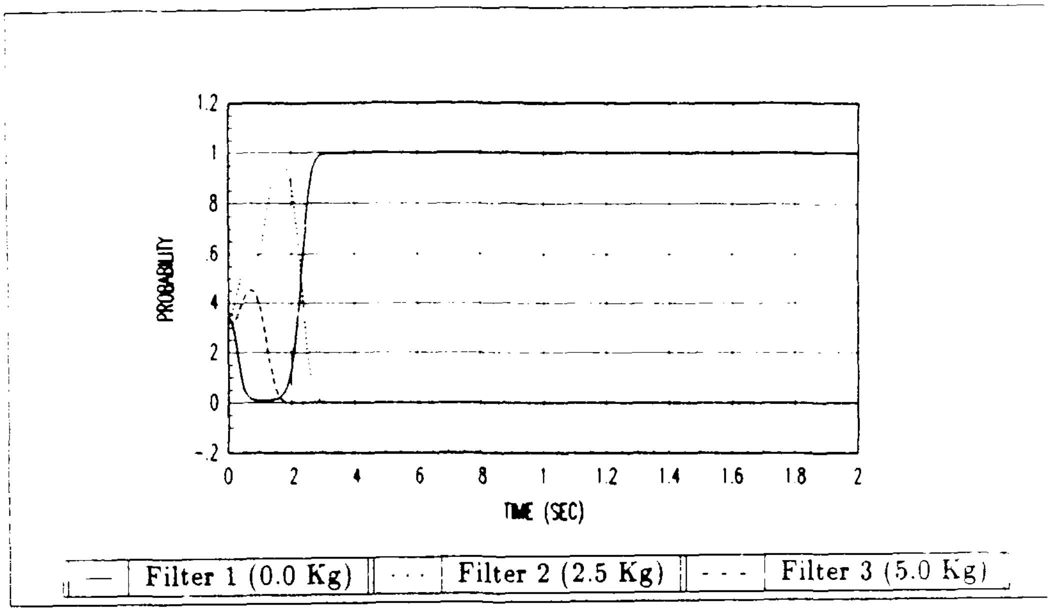


Figure C.25. Payload 0.0 Kg: Probability Profiles $\mathbf{Q} = 1$, $\mathbf{R} = 10^{-6}$ and $p_k(t_{i-1})$ not reset to 1/3

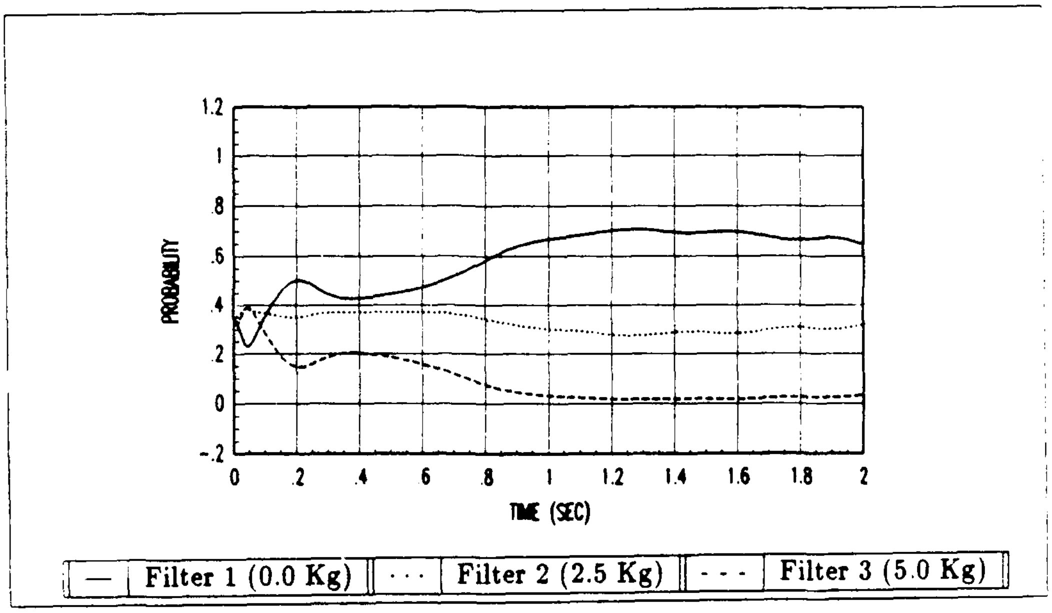


Figure C.26. Payload 0.0 Kg: Probability Profiles $\mathbf{Q} = 1$, $\mathbf{R} = 10^{-6}$ and $p_k(t_{i-1})$ reset to 1/3

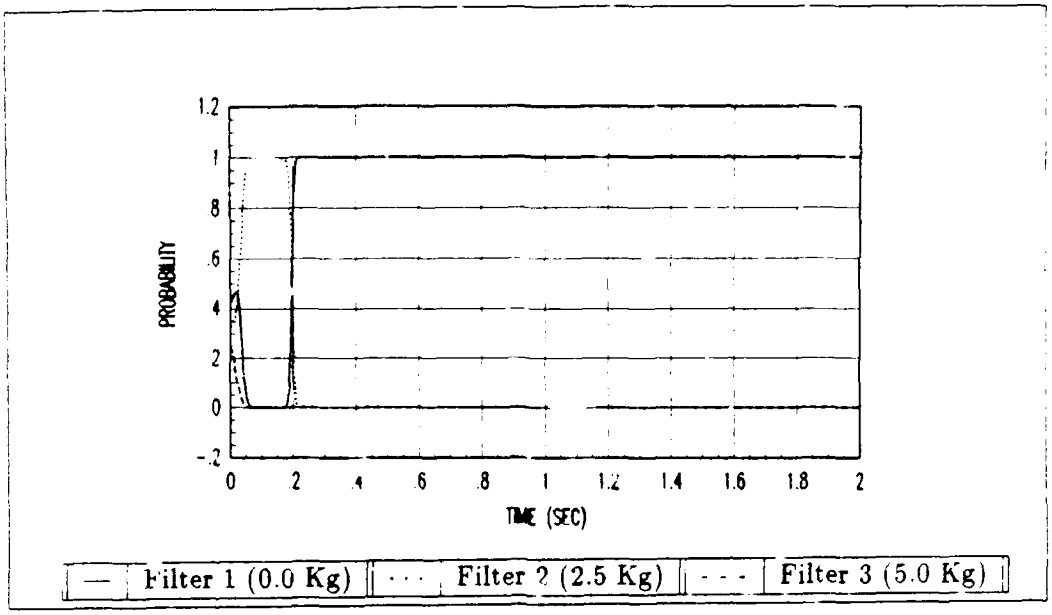


Figure C.27. Payload 0.0 Kg: Probability Profiles **Q** = 0.1, **R** = 10^{-6} and $p_k(t_{i-1})$ not reset to 1/3

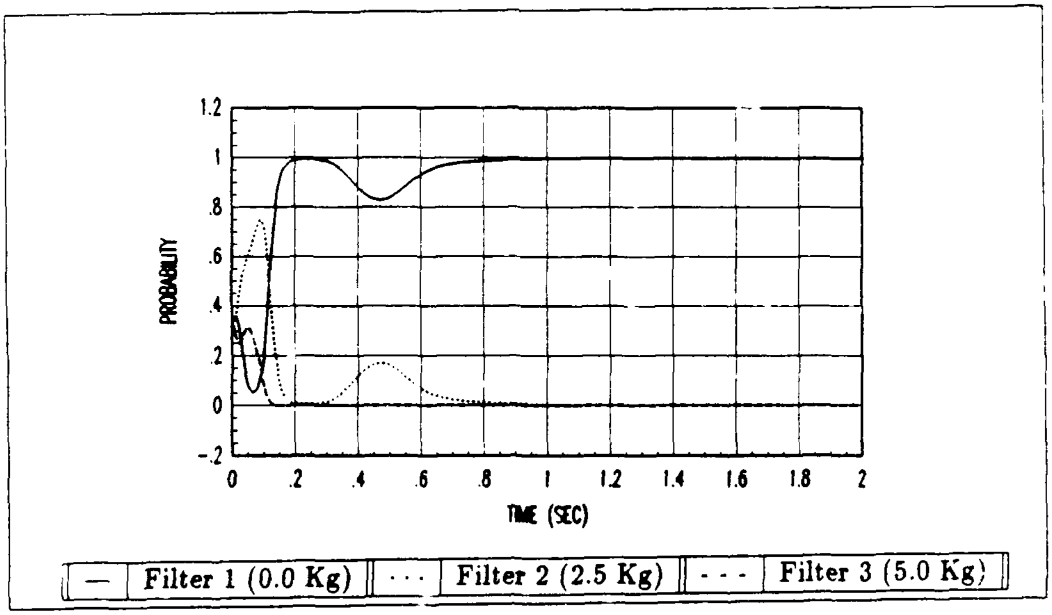


Figure C.28. Payload 0.0 Kg: Probability Profiles **Q** = 0.1, **R** = 10^{-6} and $p_k(t_{i-1})$ reset to 1/3

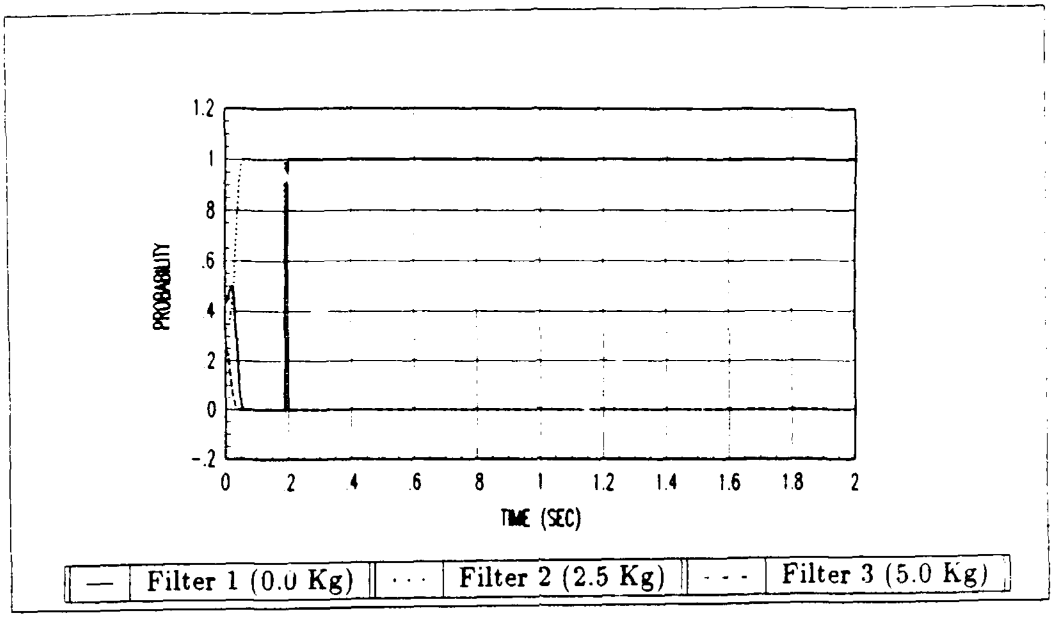


Figure C.29. Payload 0.0 Kg: Probability Profiles **Q** = 0.01, **R** = 10^{-6} and $p_k(t_{i-1})$ not reset to 1/3

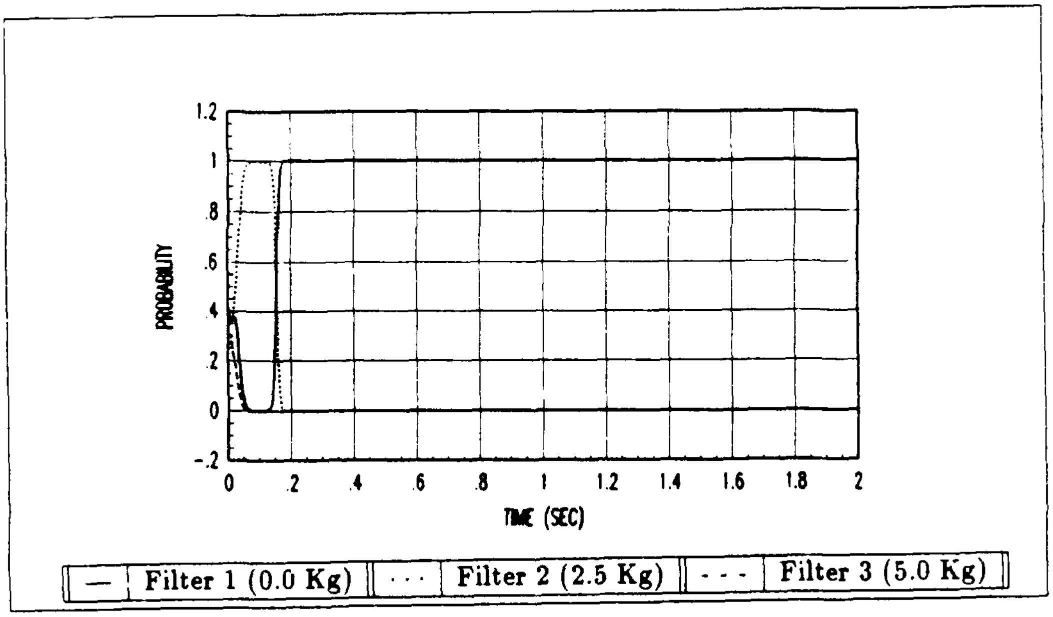


Figure C.30. Payload 0.0 Kg: Probability Profiles **Q** = 0.01, **R** = 10^{-6} and $p_k(t_{i-1})$ reset to 1/3

C.2 Payload Estimates, Probabilities and Error Profiles

The plots of payload estimates, hypothesis conditional probabilities, and tracking error performance are contained in this section. These plots are a result of the evaluation conducted for external payloads of 5.0, 4.0, 3.0, 2.5, 2.0, 1.0, and 0.0 Kg using Trajectory One. The plots are arranged in the following order for each external payload case:

- payload estimate plot,
- corresponding hypothesis probabilities plot, and
- the tracking error performance plot for the three links.

The tracking error plots include the tracking error profiles of a single model-based control (SMBC) algorithm with and without knowledge of the external payload. The tracking performance of the informed SMBC is the best possible performance that the OL/MMBC algorithm could expect to achieve and was used as a benchmark.

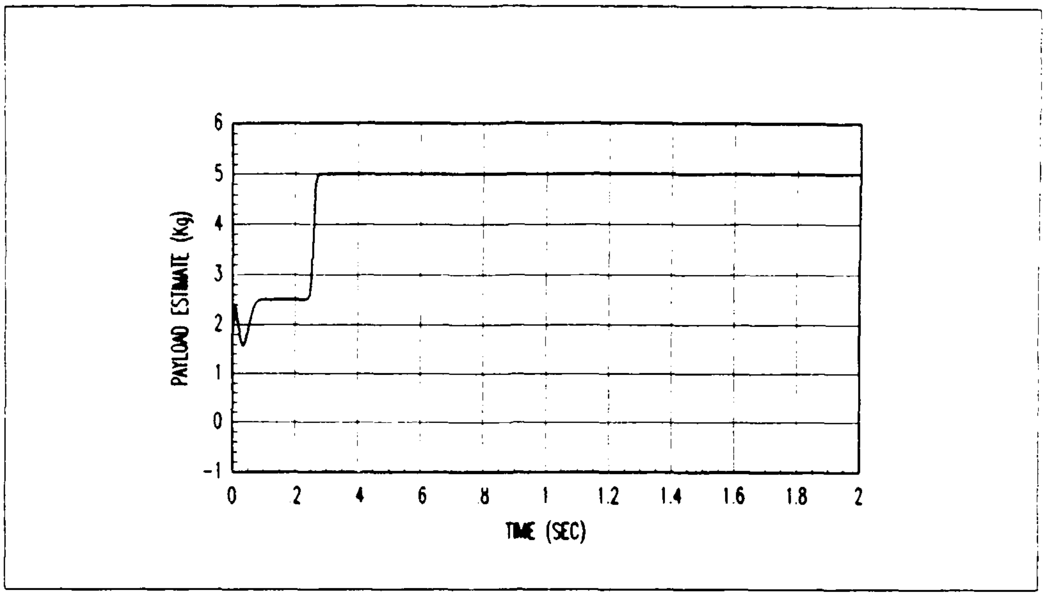


Figure C.31. Payload 5.0 Kg: MMAE Payload Estimate

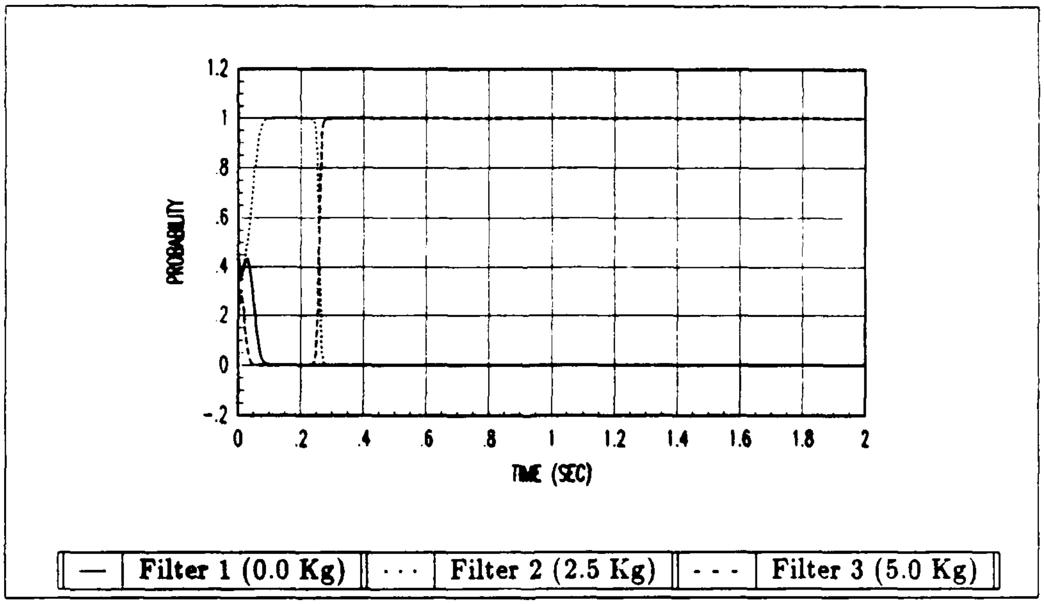


Figure C.32. Payload 5.0 Kg: Probability Profiles

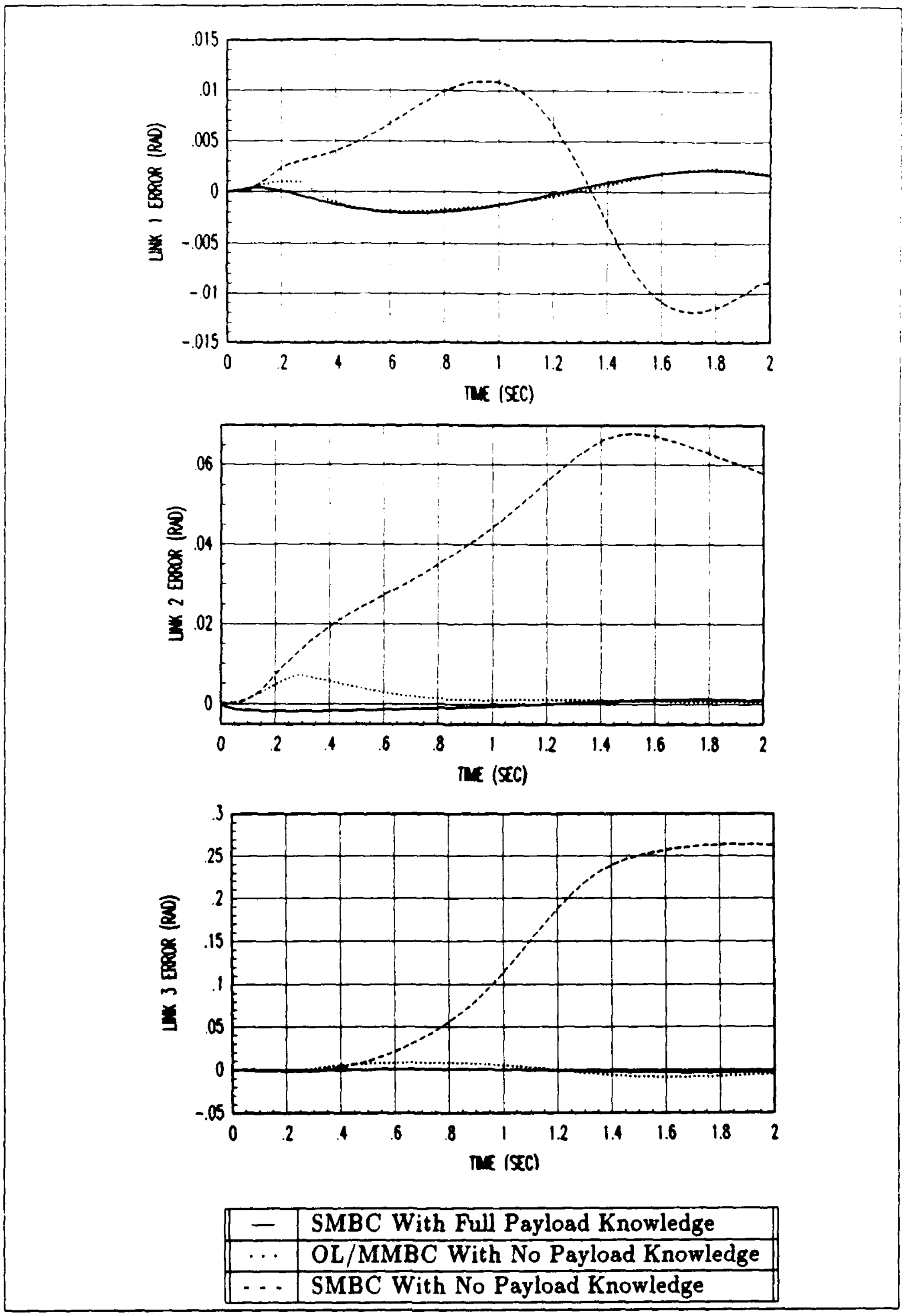


Figure C.33. Payload 5.0 Kg: Trajectory One Tracking Error

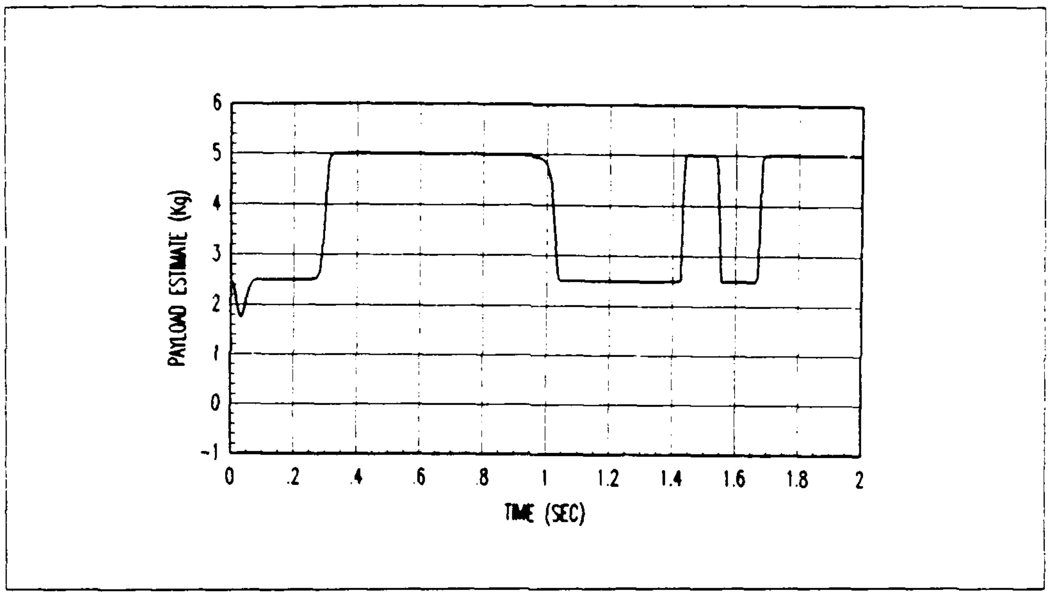


Figure C.34. Payload 4.0 Kg: MMAE Payload Estimate

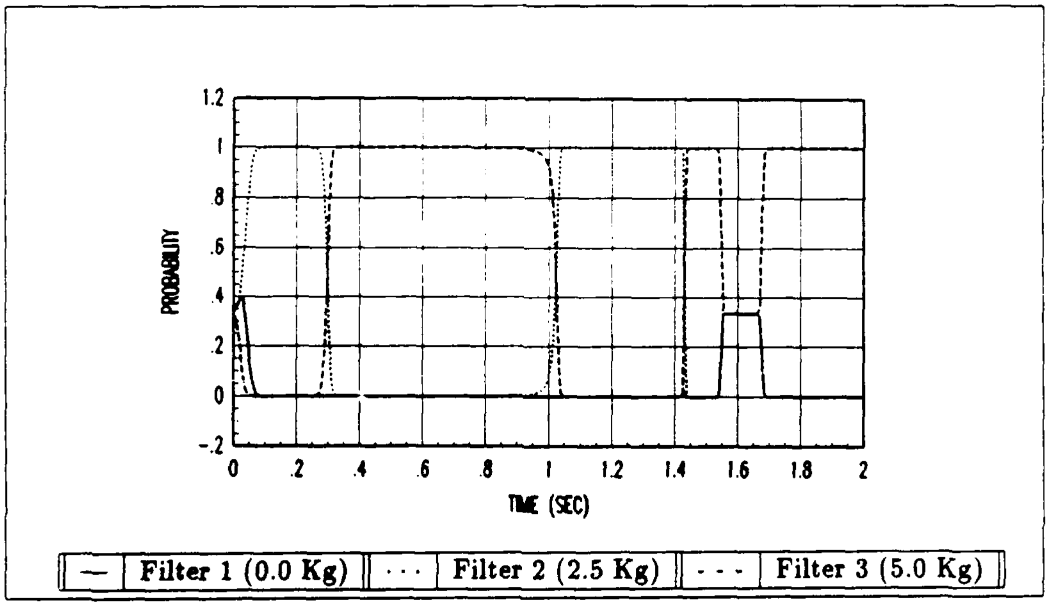


Figure C.35. Payload 4.0 Kg: Probability Profiles

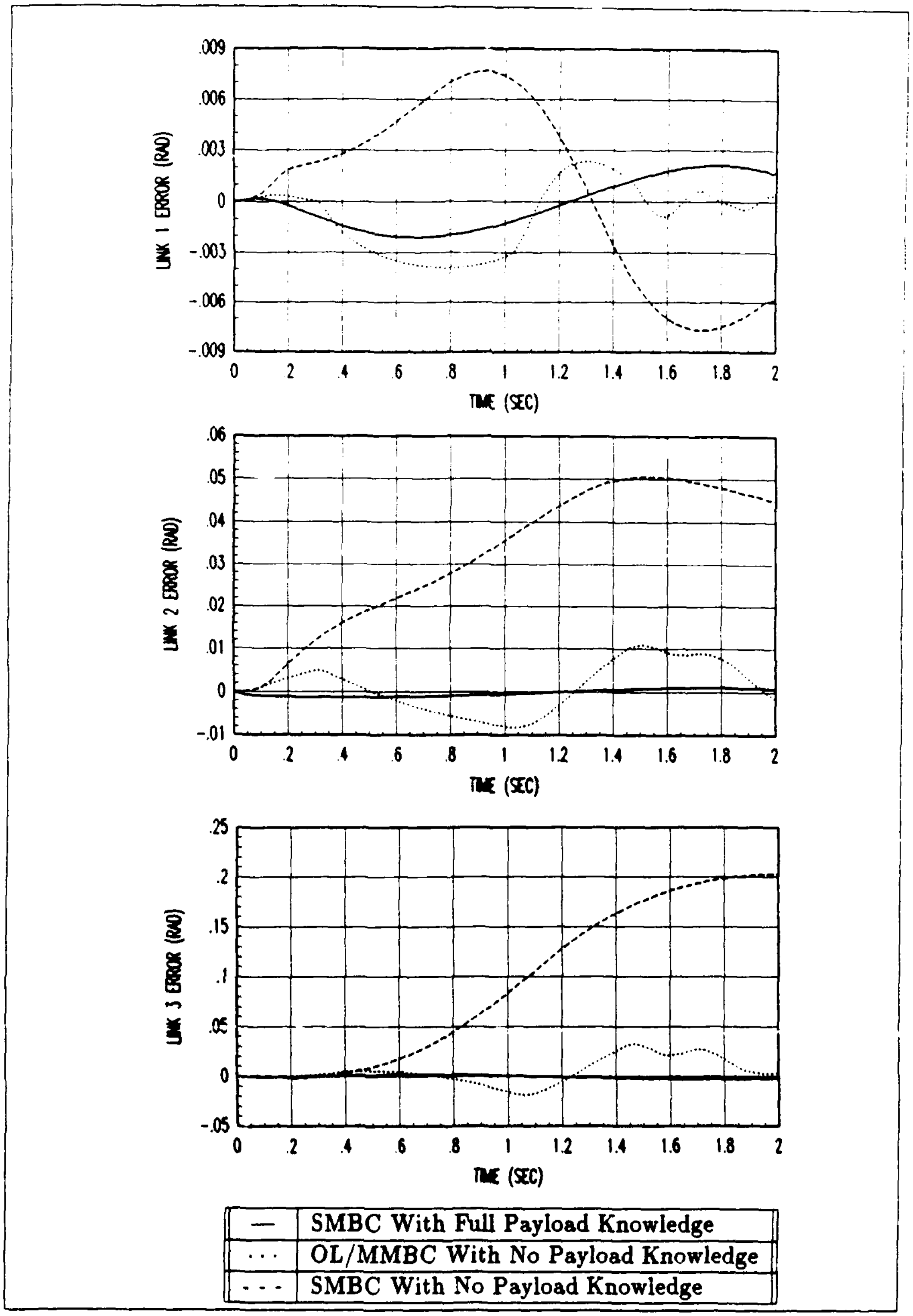


Figure C.36. Payload 4.0 Kg: Trajectory One Tracking Error

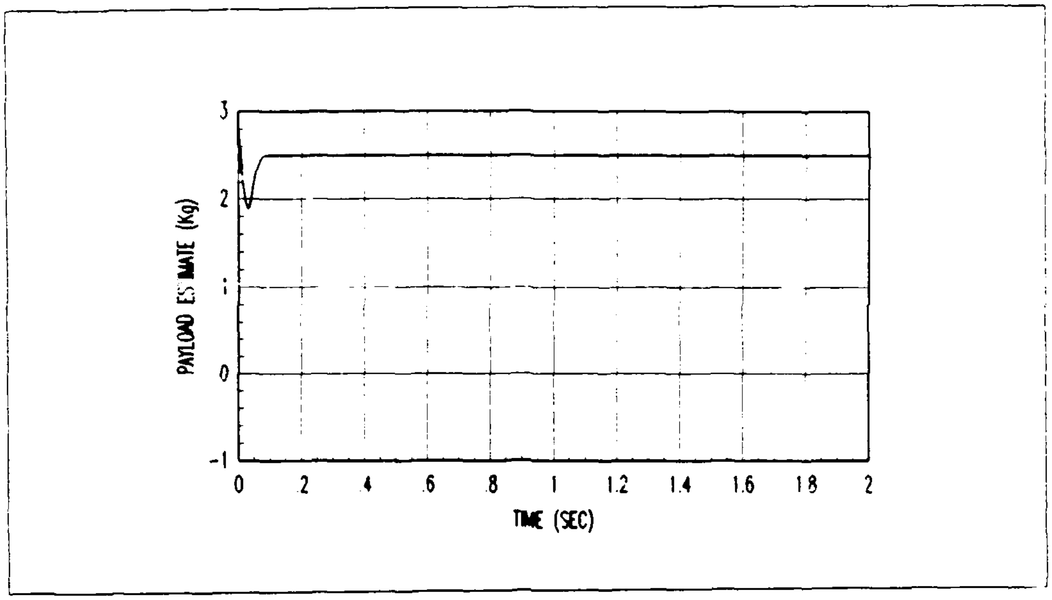


Figure C.37. Payload 3.0 Kg: MMAE Payload Estimate

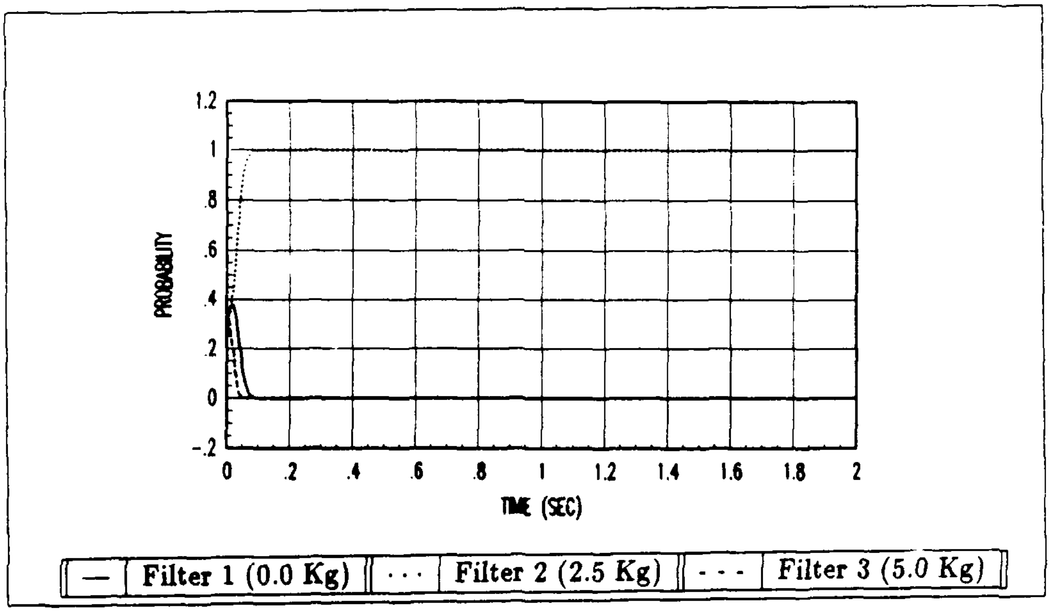


Figure C.38. Payload 3.0 Kg: Probability Profiles

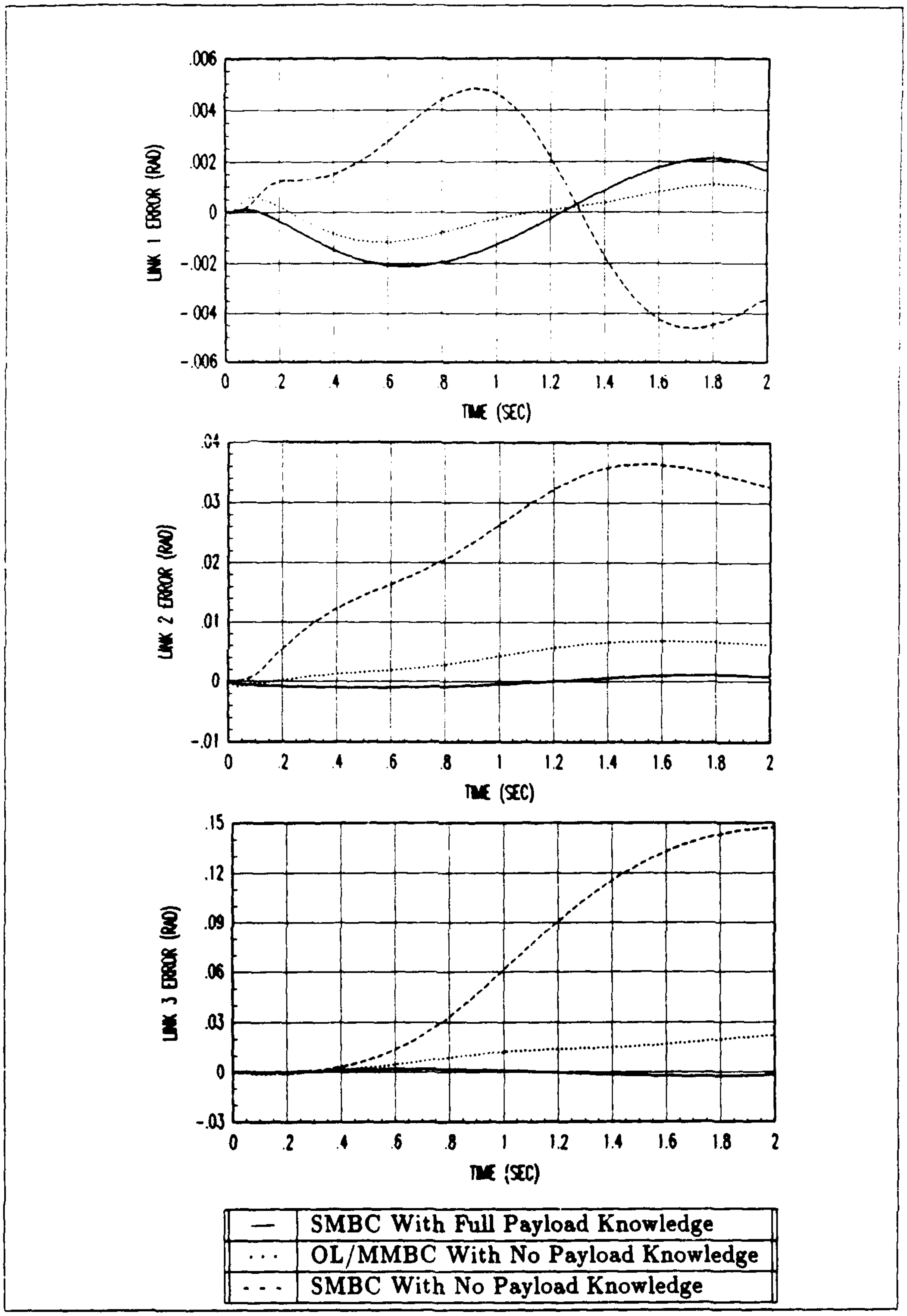


Figure C.39. Payload 3.0 Kg: Trajectory One Tracking Error

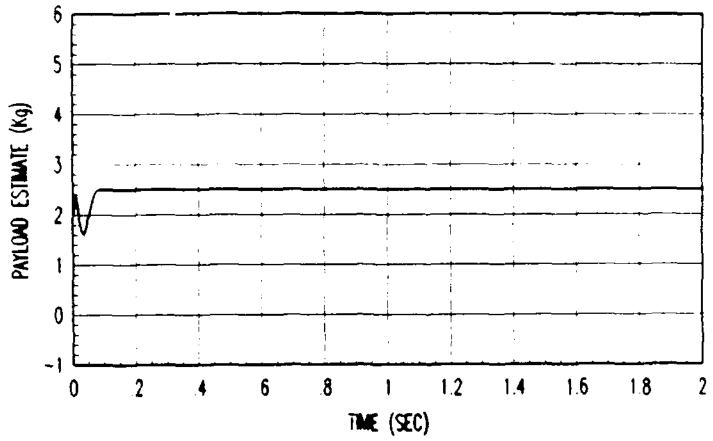


Figure C.40. Payload 2.5 Kg: MMAE Payload Estimate

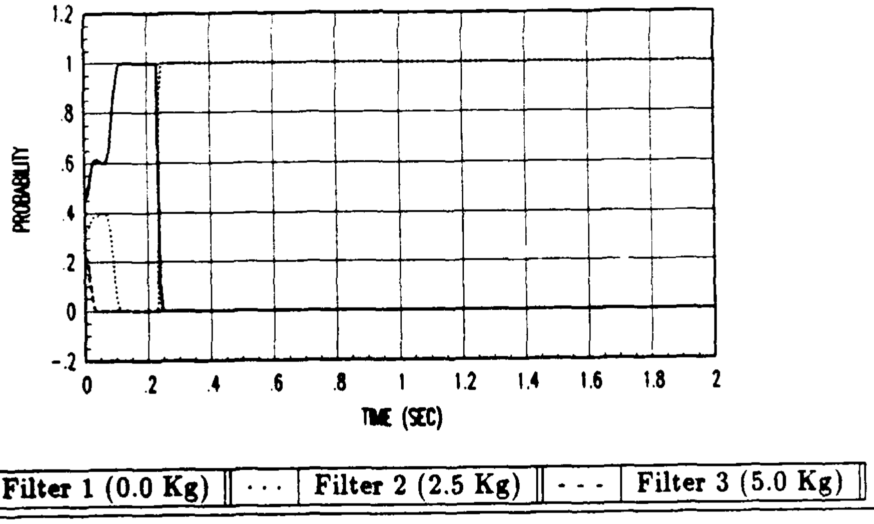


Figure C.41. Payload 2.5 Kg: Probability Profiles

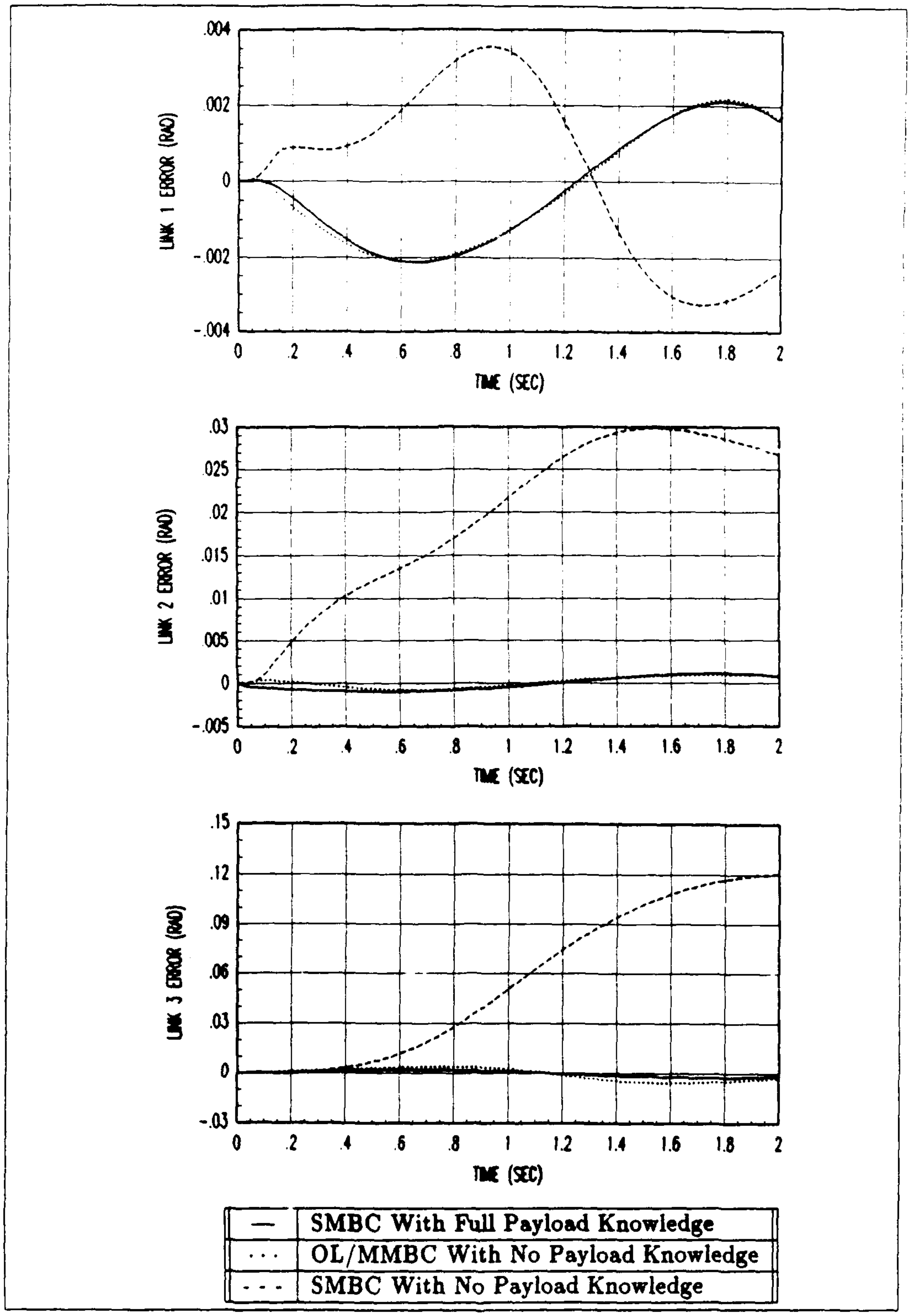


Figure C.42. Payload 2.5 Kg: Trajectory One Tracking Error

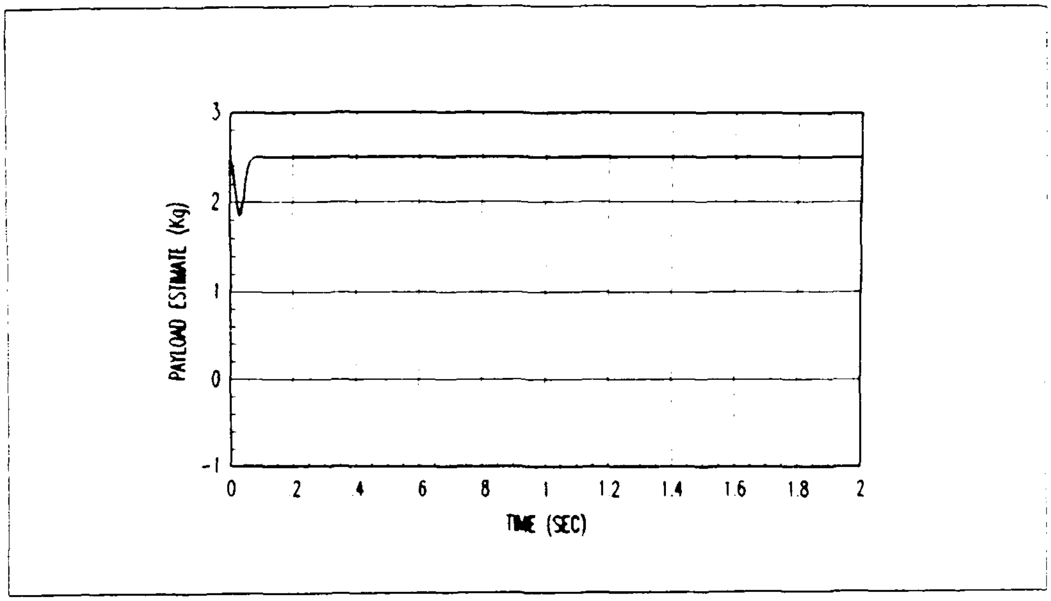


Figure C.43. Payload 2.0 Kg: MMAE Payload Estimate

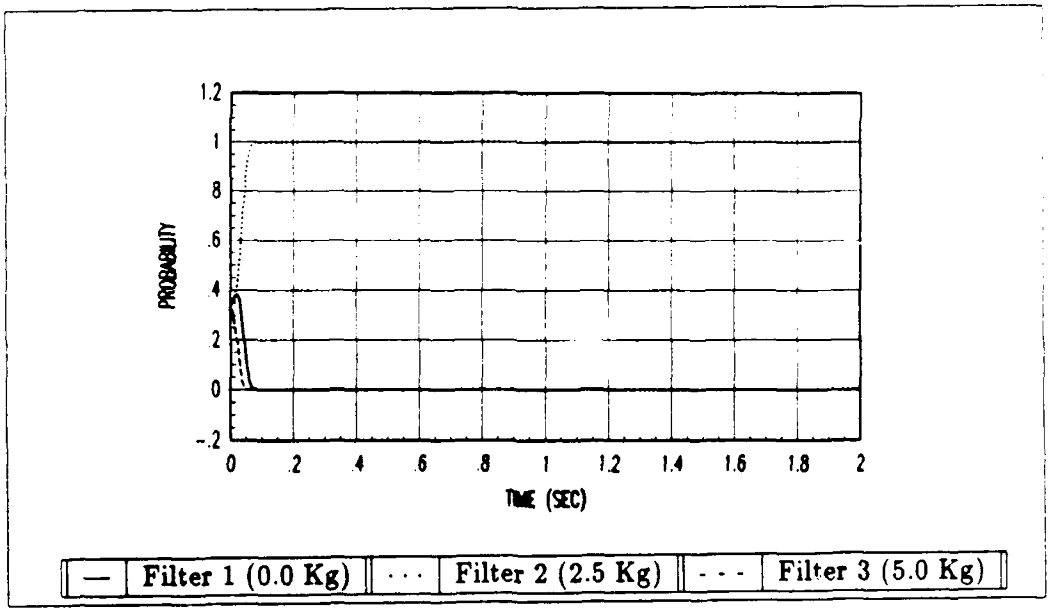


Figure C.44. Payload 2.0 Kg: Probability Profiles

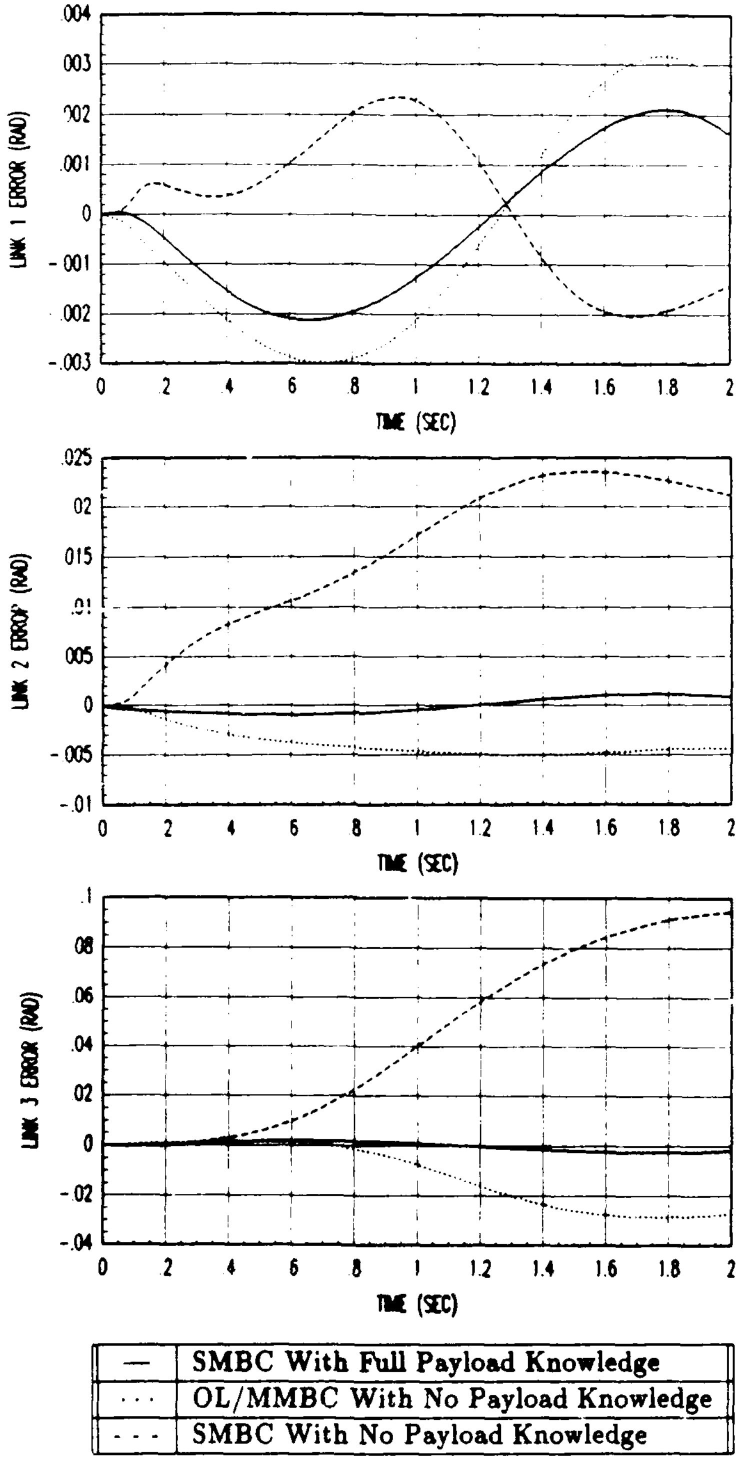


Figure C.45. Payload 2.0 Kg: Trajectory One Tracking Error

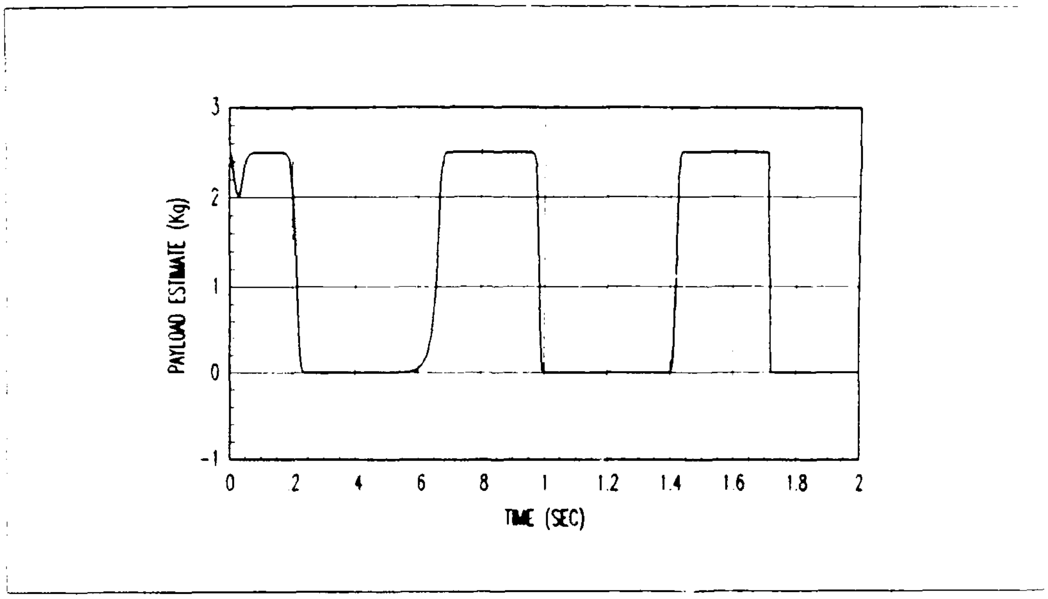


Figure C.46. Payload 1.0 Kg: MMAE Payload Estimate

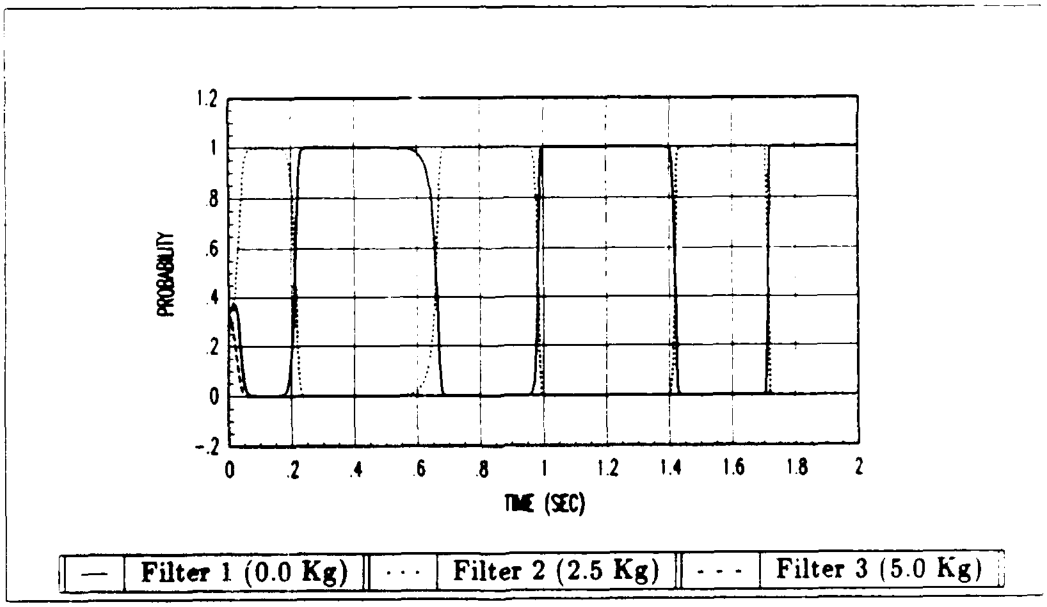


Figure C.47. Payload 1.0 Kg: Probability Profiles

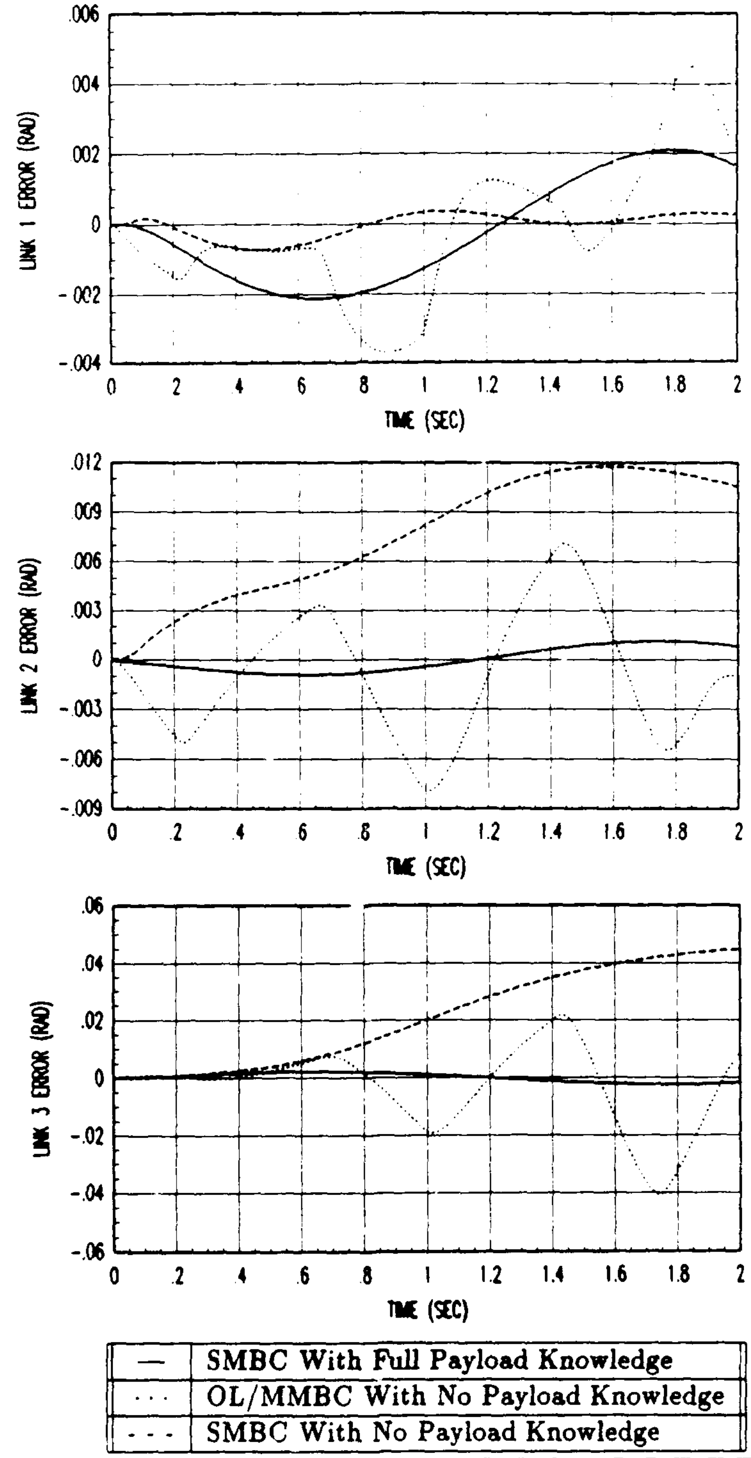


Figure C.48. Payload 1.0 Kg: Trajectory One Tracking Error

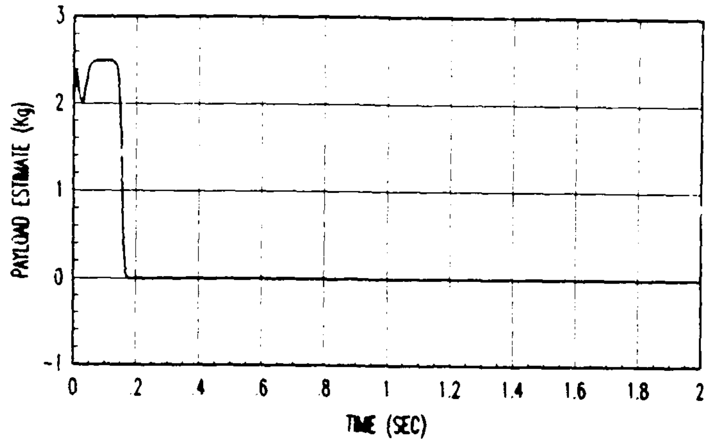


Figure C.49. Payload 0.0 Kg: MMAE Payload Estimate

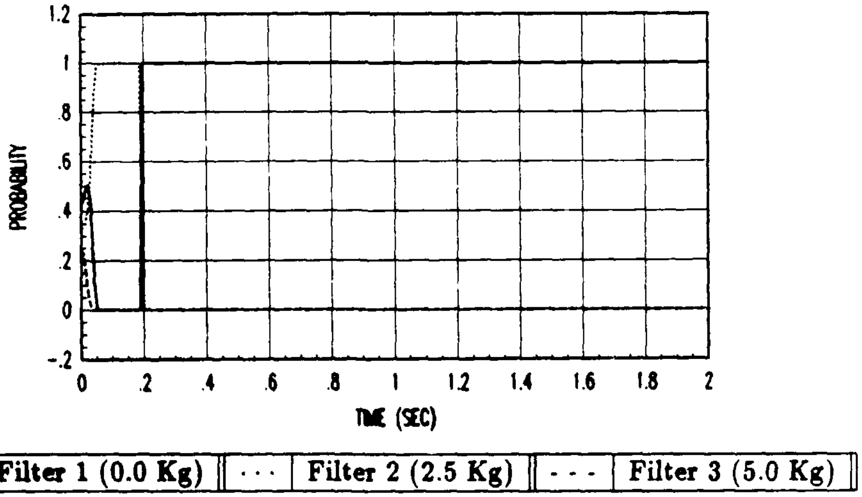


Figure C.50. Payload 0.0 Kg: Probability Profiles

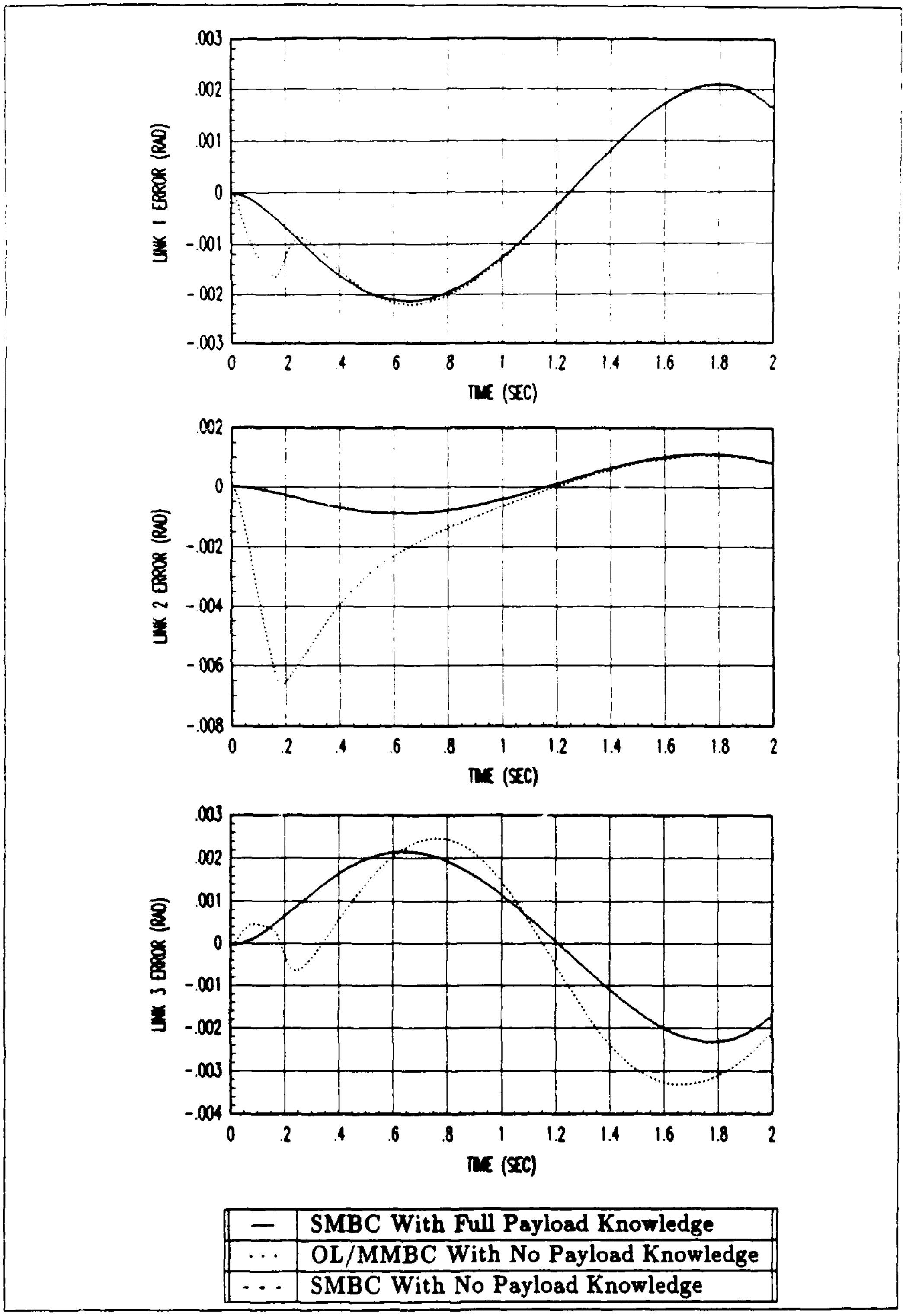


Figure C.51. Payload 0.0 Kg: Trajectory One Tracking Error

Appendix D. *OL/MMBC Experimental Evaluation Plots*

This appendix contains the plots of the OL/MMBC experimental evaluations on a PUMA-560 manipulator for Trajectory One using external payloads of 3.0 and 0.0 Kg. The plots included in this appendix are for the soft and high PD gains shown in Table 5.1.

D.1 Results Using Soft PD Gains

The plots of payload estimates and tracking errors for the 3.0 and 0.0 Kg payloads for Trajectory One using the soft PD gains of Table 5.1 are contained in this section. The plots are listed in the following order for each payload

- OL/MMBC mean plus one-sigma tracking errors,
- OL/MMBC, informed, and uninformed SMBC mean tracking errors, and
- MMAE mean payload estimate.

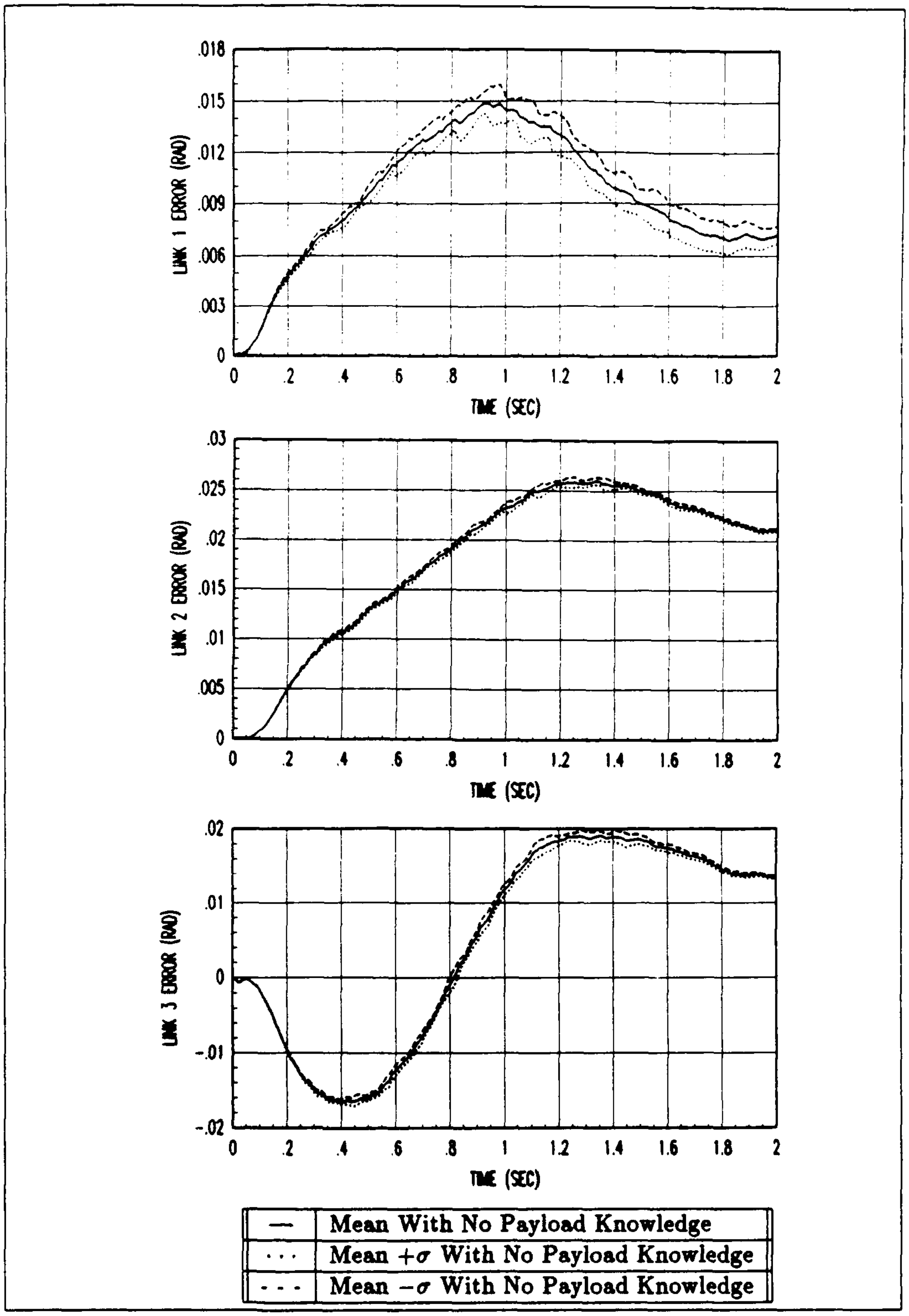


Figure D.1. Payload 3.0 Kg: OL/MMBC Tracking Errors for Trajectory One Using Soft Gains

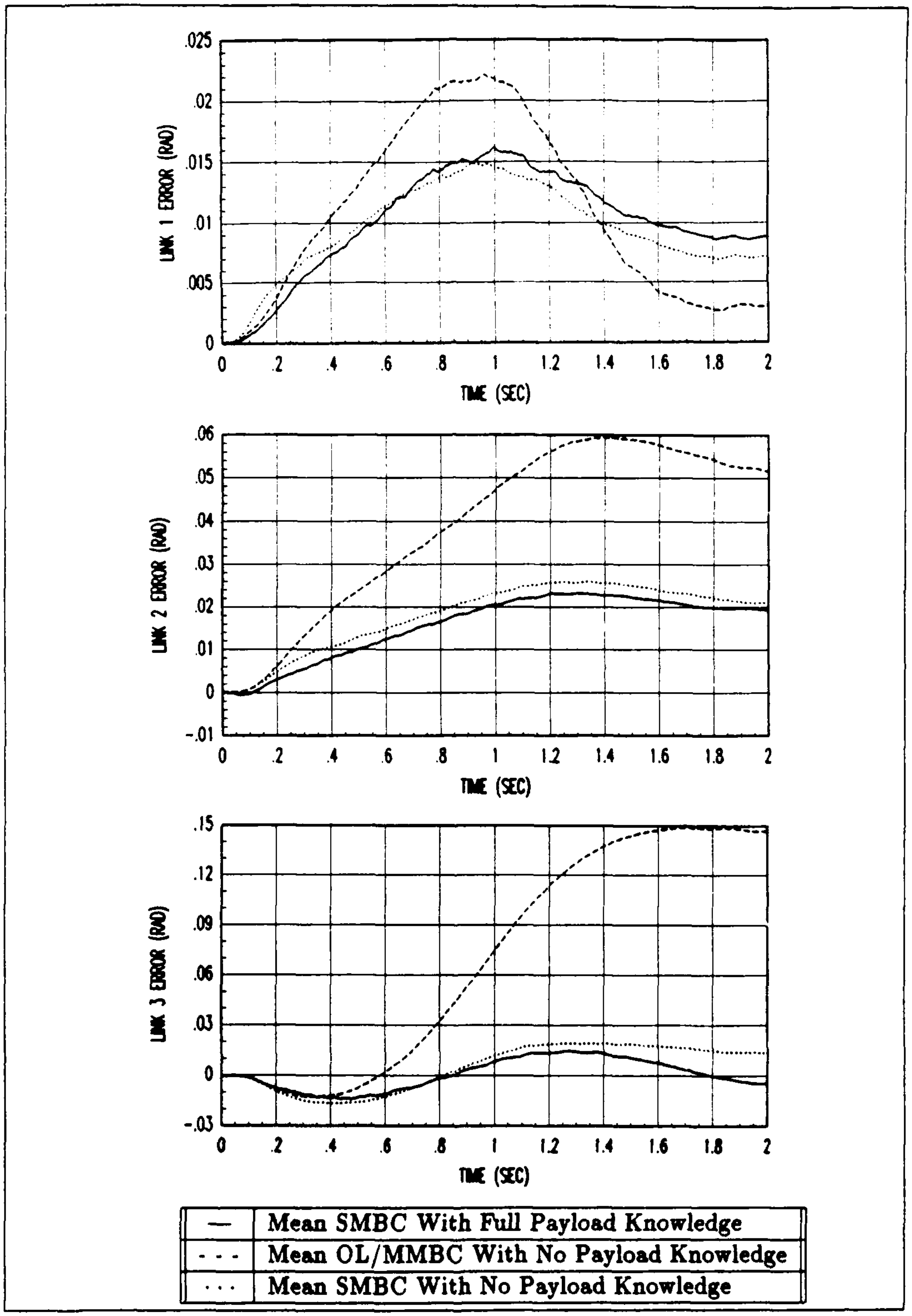


Figure D.2. Payload 3.0 Kg: Tracking Errors for Trajectory One Using Soft Gains

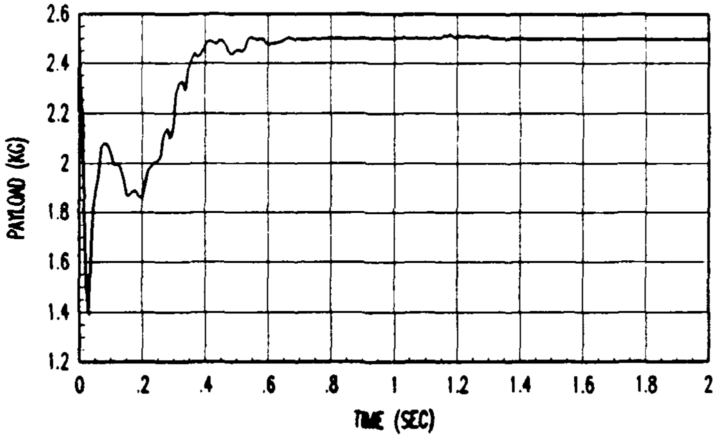
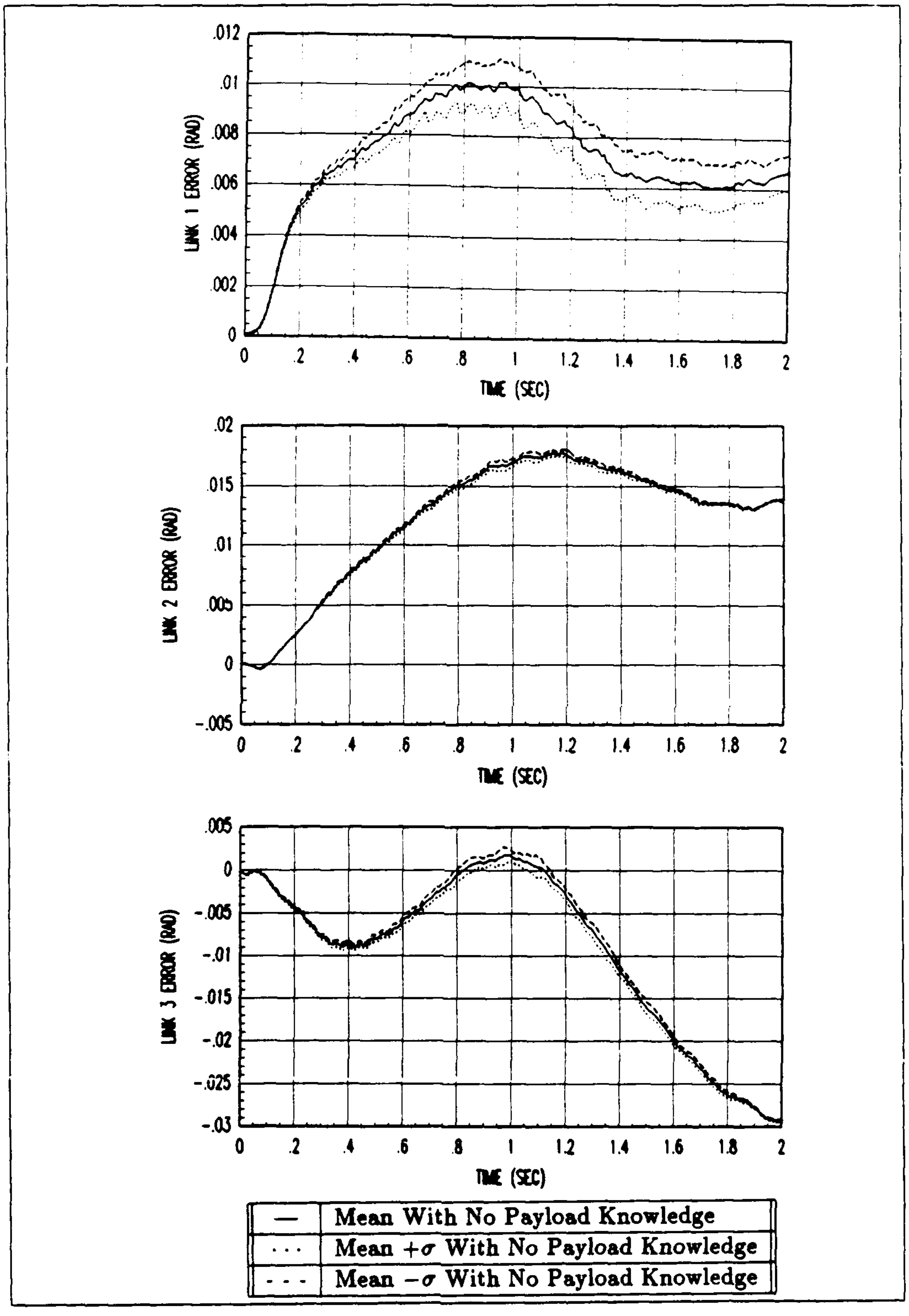


Figure D.3. Payload 3.0 Kg: Mean MMAE Payload Estimate Using Soft Gains



Fi ;ure D.4. Payload 2.0 Kg: OL/MMBC Tracking Errors for Trajectory One Using Soft Gains

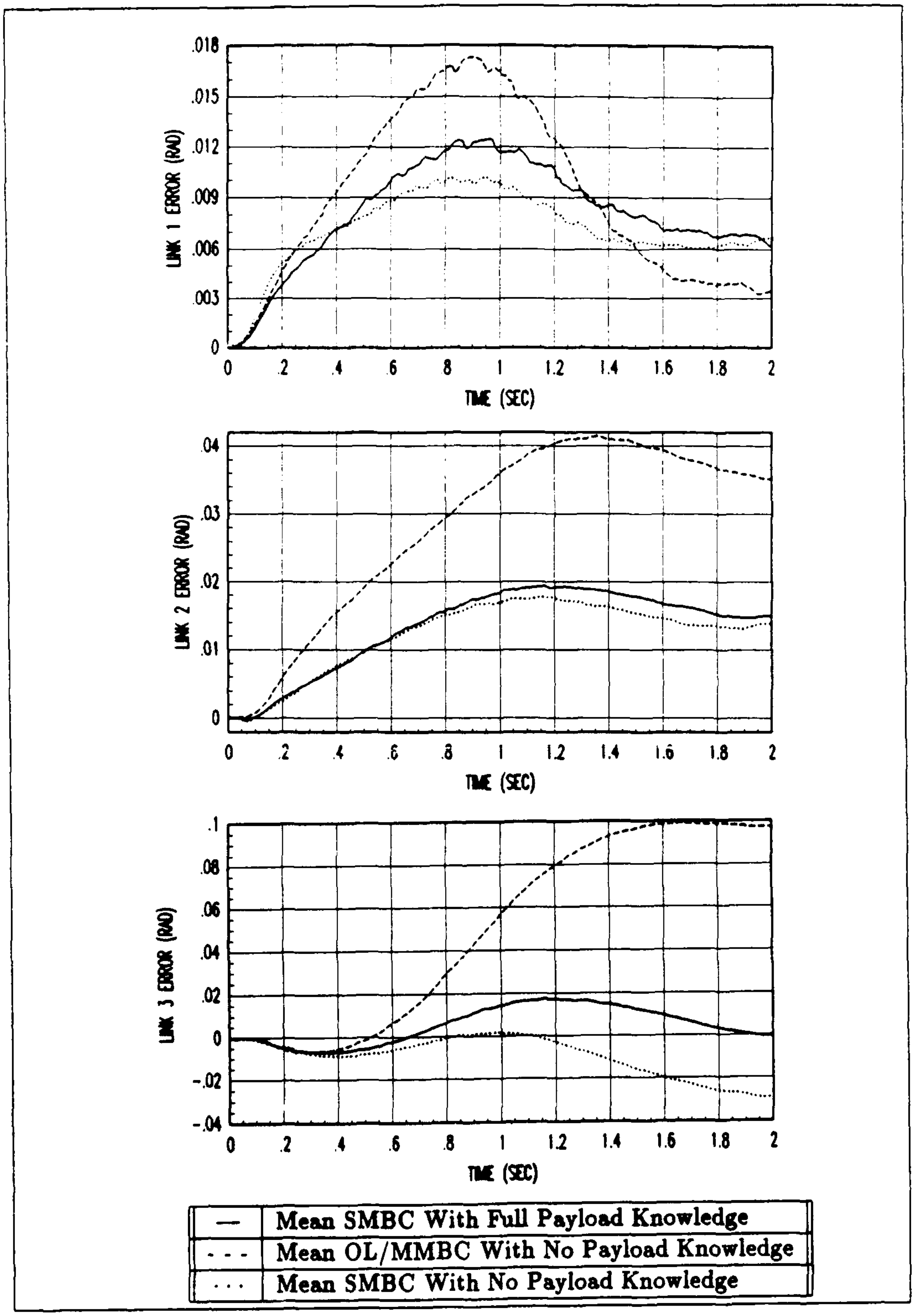


Figure D.5. Payload 2.0 Kg: Tracking Errors for Trajectory One Using Soft Gains

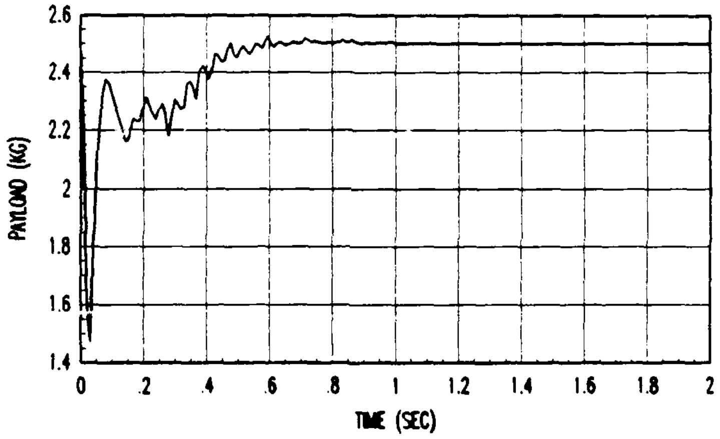


Figure D.6. Payload 2.0 Kg: Mean MMAE Payload Estimate Using Soft Gains

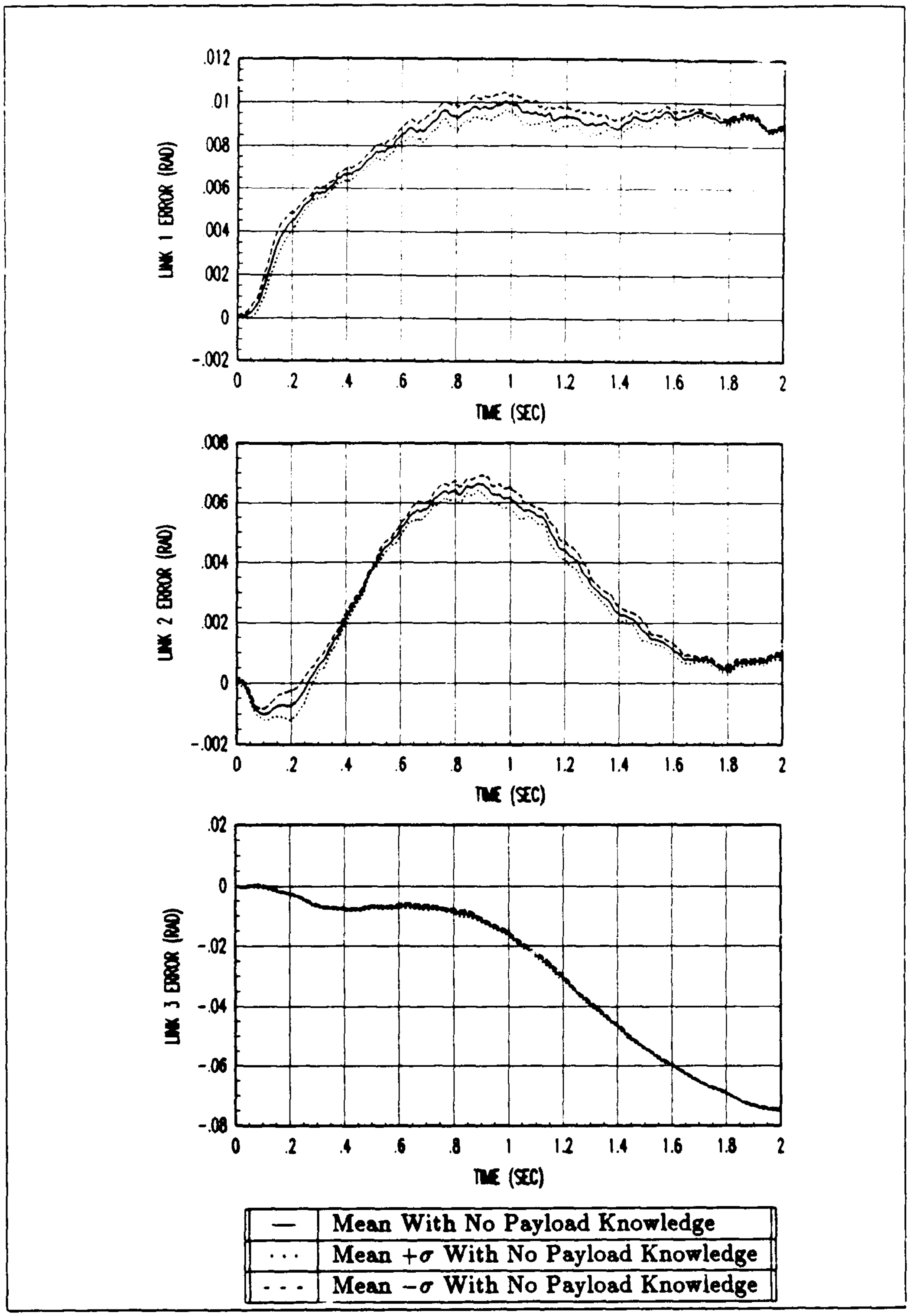


Figure D.7. Payload 1.0 Kg: OL/MMBC Tracking Errors for Trajectory One Using Soft Gains

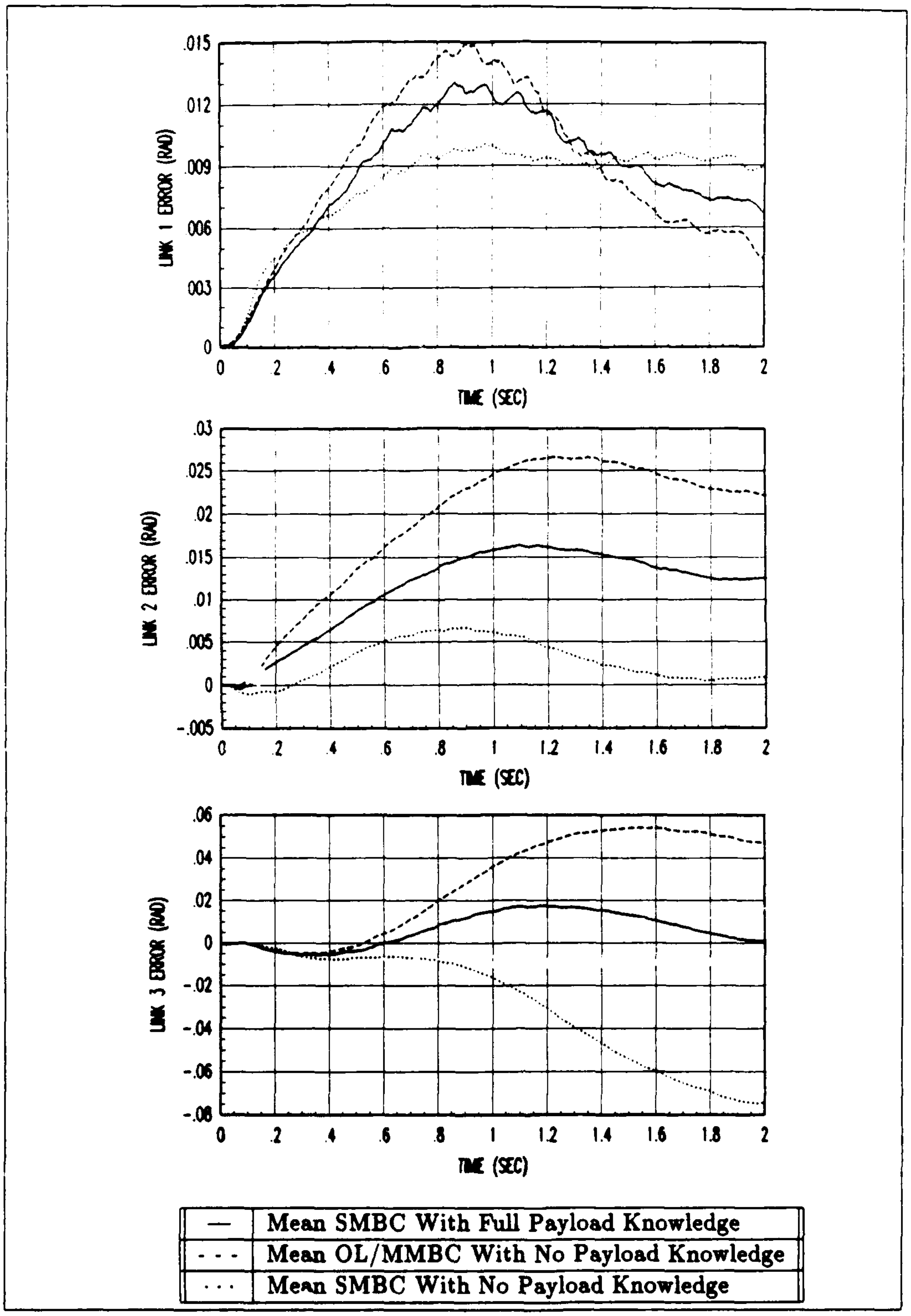


Figure D.8. Payload 1.0 Kg: Tracking Errors for Trajectory One Using Soft Gains

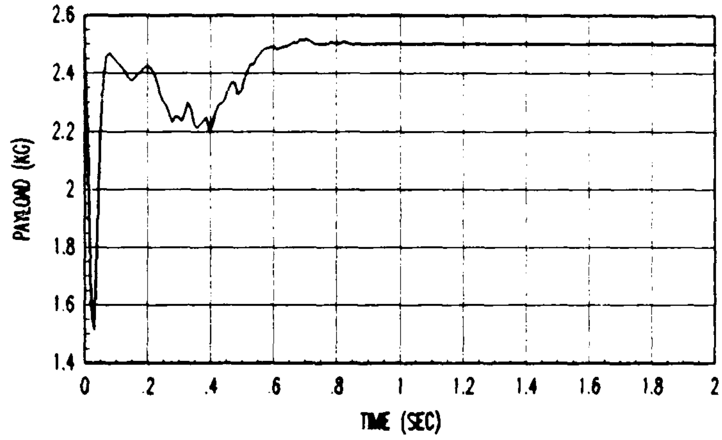


Figure D.9. Payload 1.0 Kg: Mean MMAE Payload Estimate Using Soft Gains

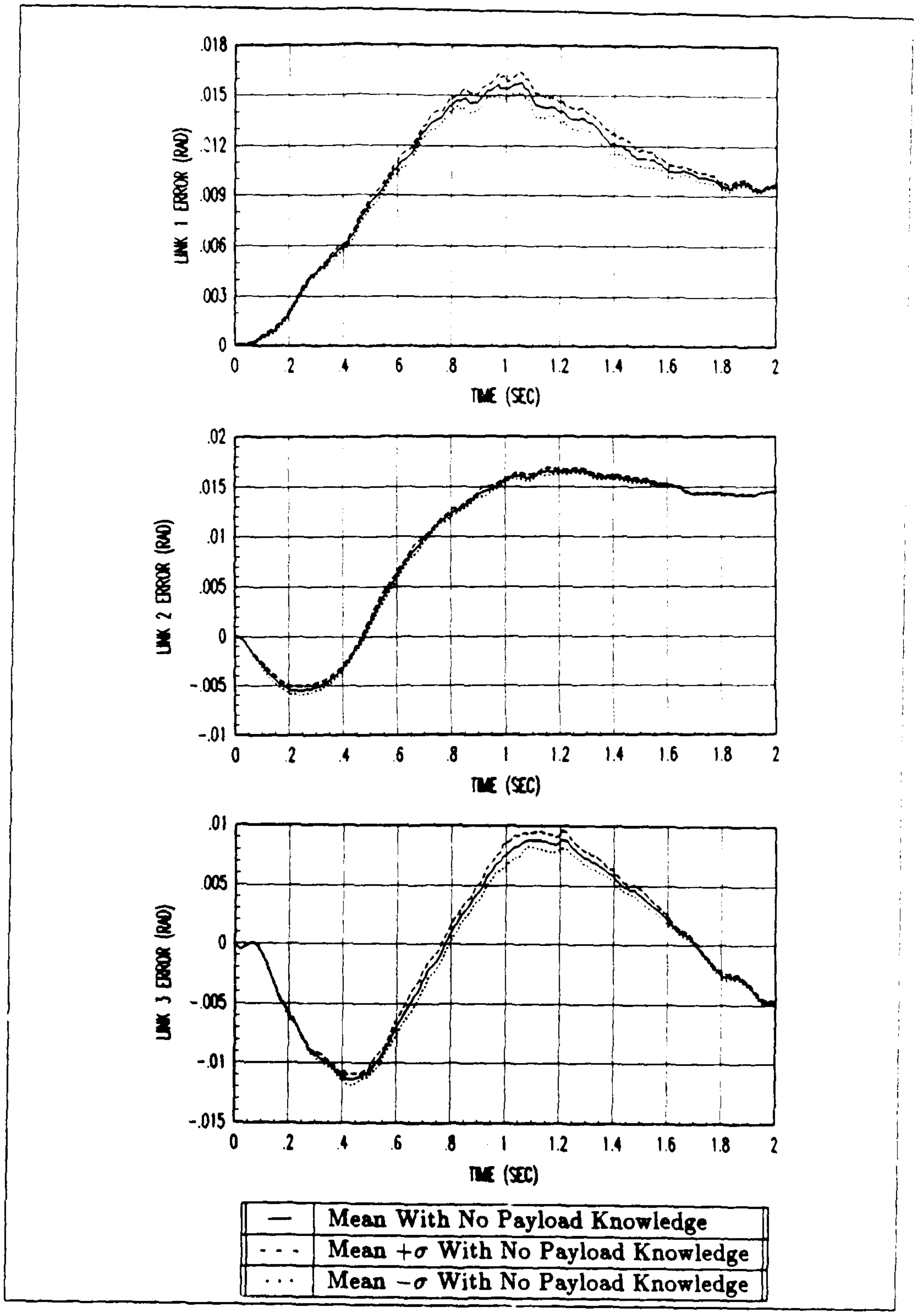


Figure D.10. Payload 0.0 Kg: OL/MMBC Tracking Errors for Trajectory One Using Soft Gains

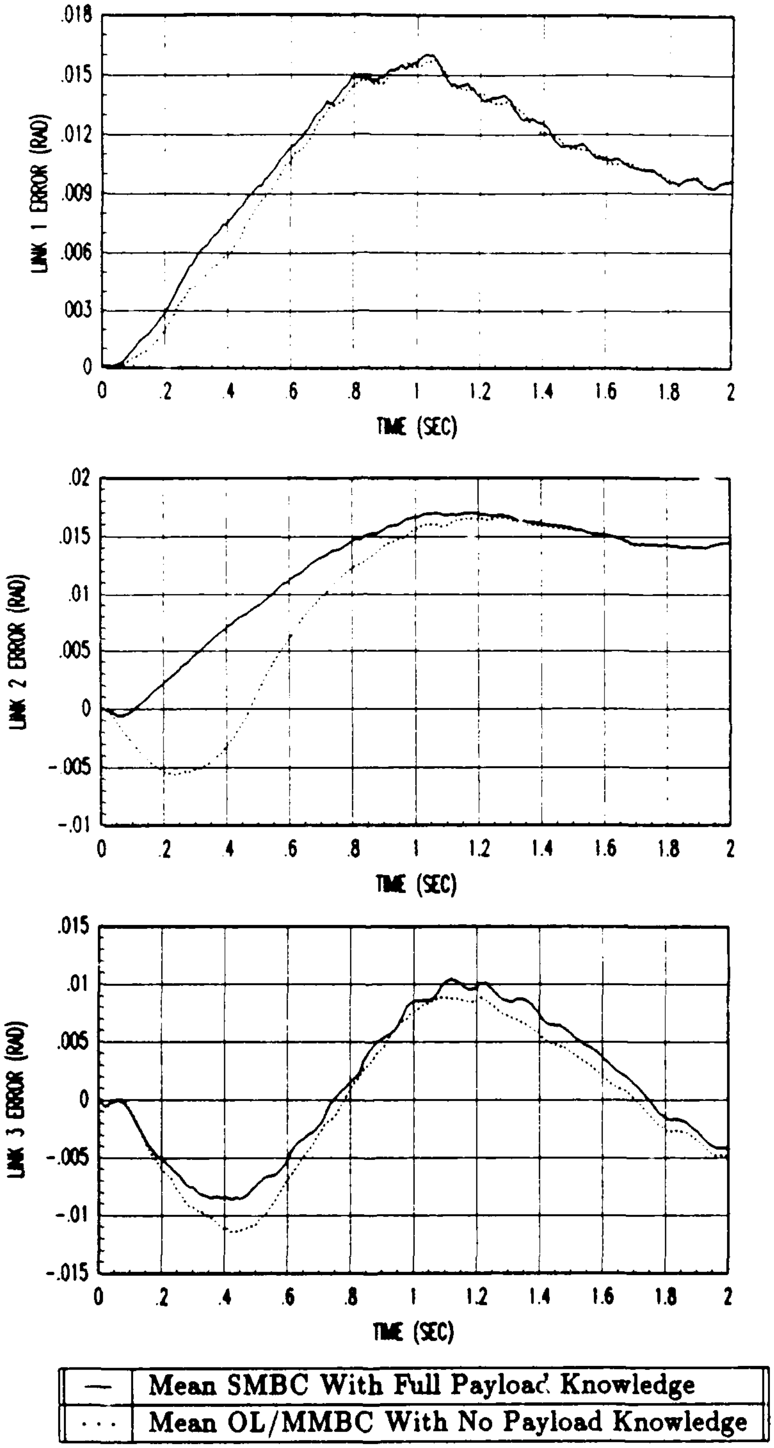


Figure D.11. Payload 0.0 Kg: Tracking Errors for Trajectory One Using Soft Gains

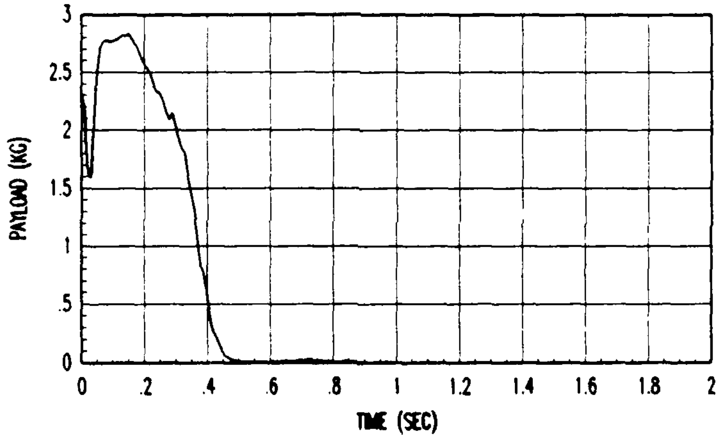


Figure D.12. Payload 0.0 Kg: Mean MMAE Payload Estimate Using Soft Gains

D.2 Results Using High PD Gains

The plots of payload estimates and tracking errors for the 3.0 and 0.0 Kg payloads for Trajectory One using the high PD gains of Table 5.1 are contained in this section. The plots are listed in the following order for each payload

- OL/MMBC mean plus one-sigma tracking errors,
- OL/MMBC, informed, and uninformed SMBC mean tracking errors, and
- MMAE mean payload estimate.

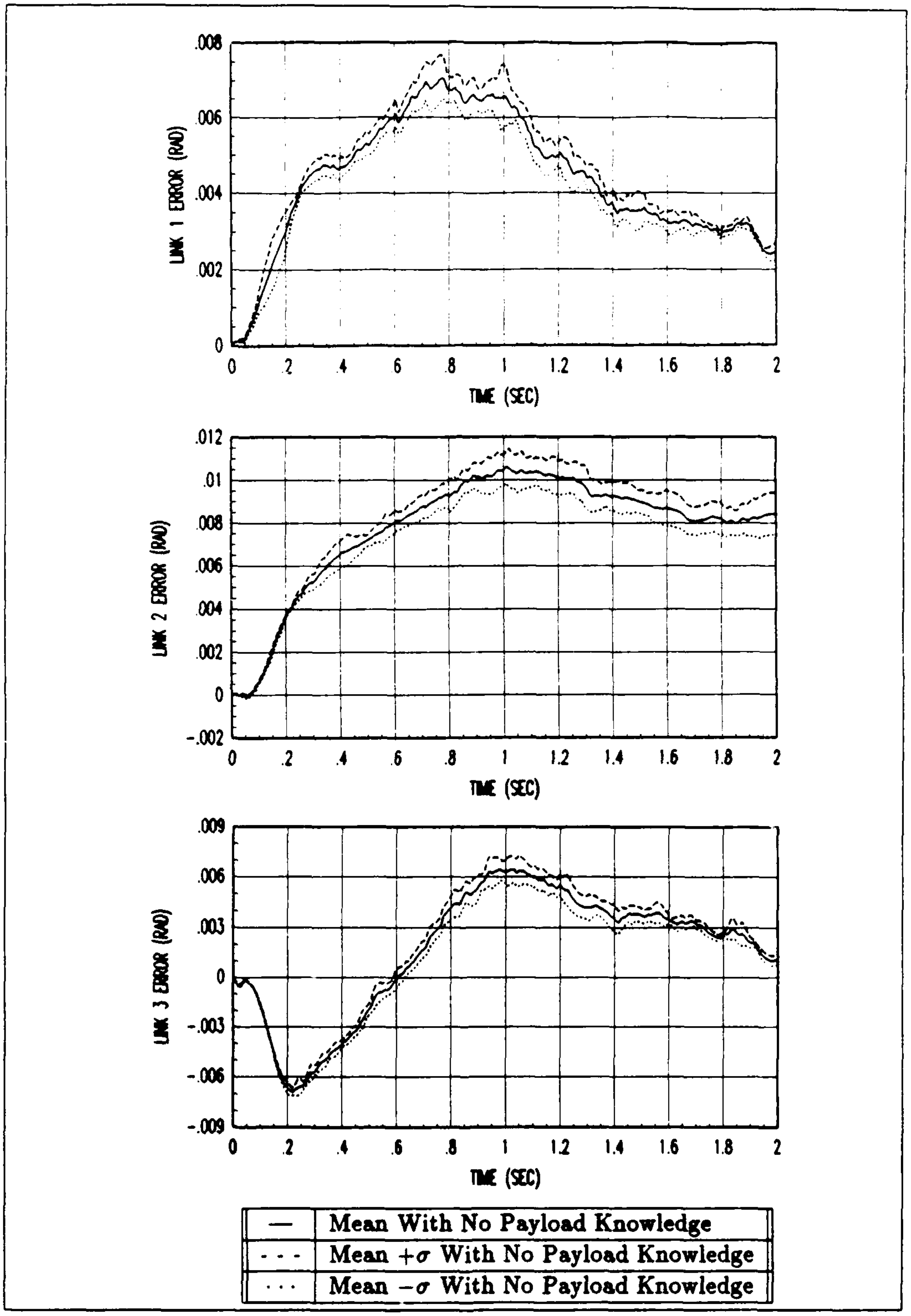


Figure D.13. Payload 3.0 Kg: OL/MMBC Tracking Errors for Trajectory One Using High Gains

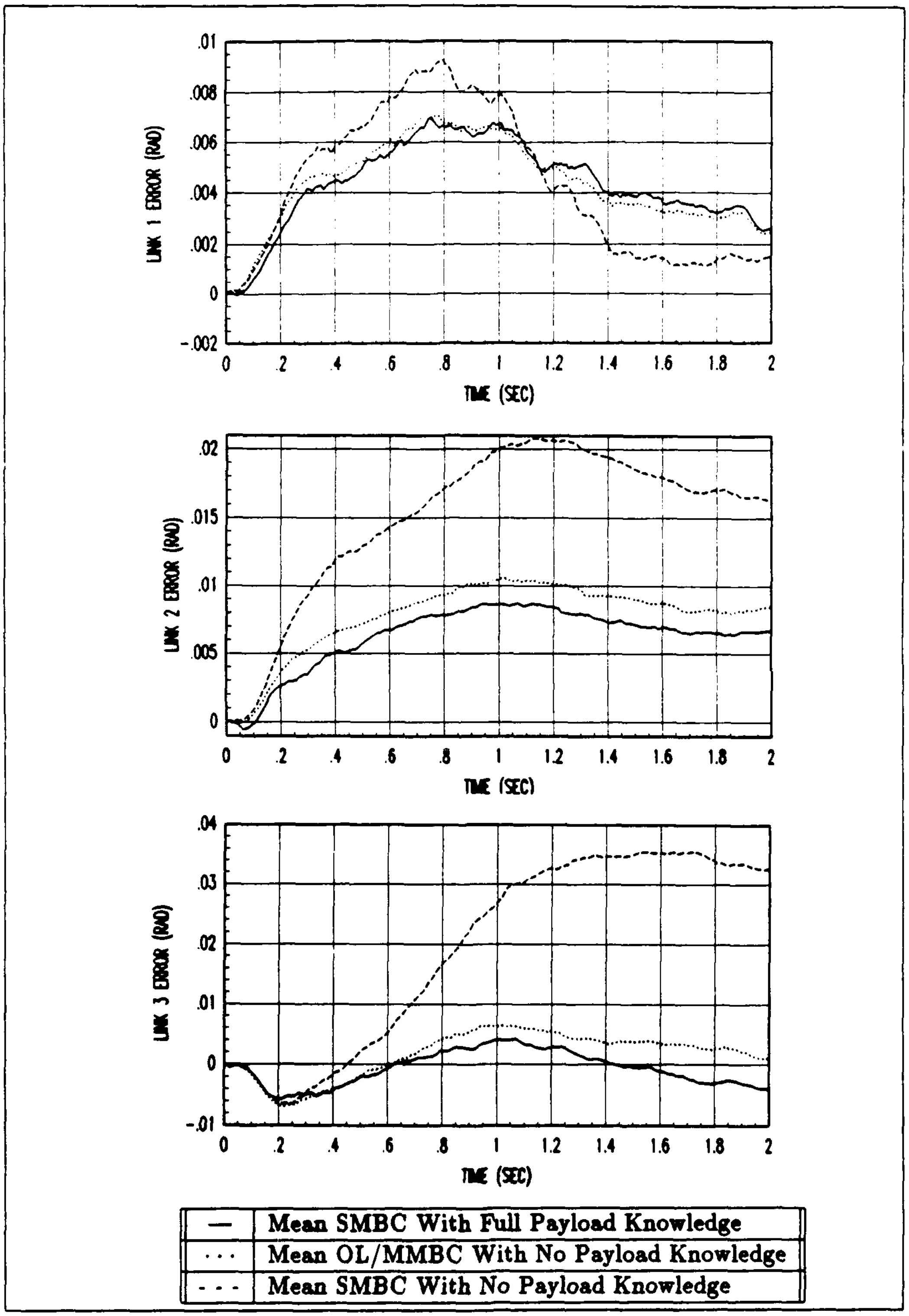


Figure D.14. Payload 3.0 Kg: Tracking Errors for Trajectory One Using High Gains

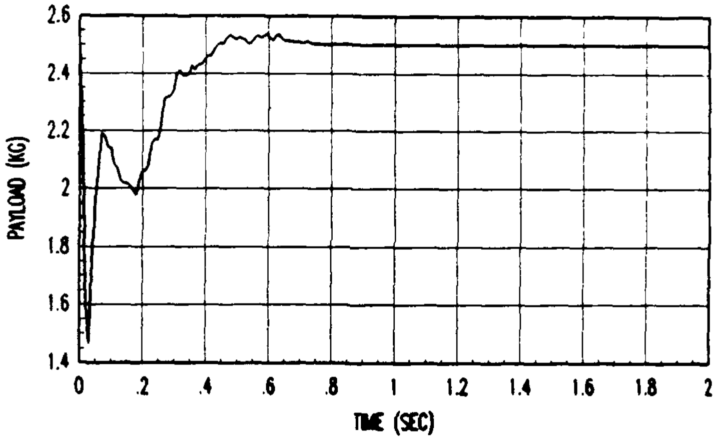


Figure D.15. Payload 3.0 Kg: Mean MMAE Payload Estimate Using High Gains

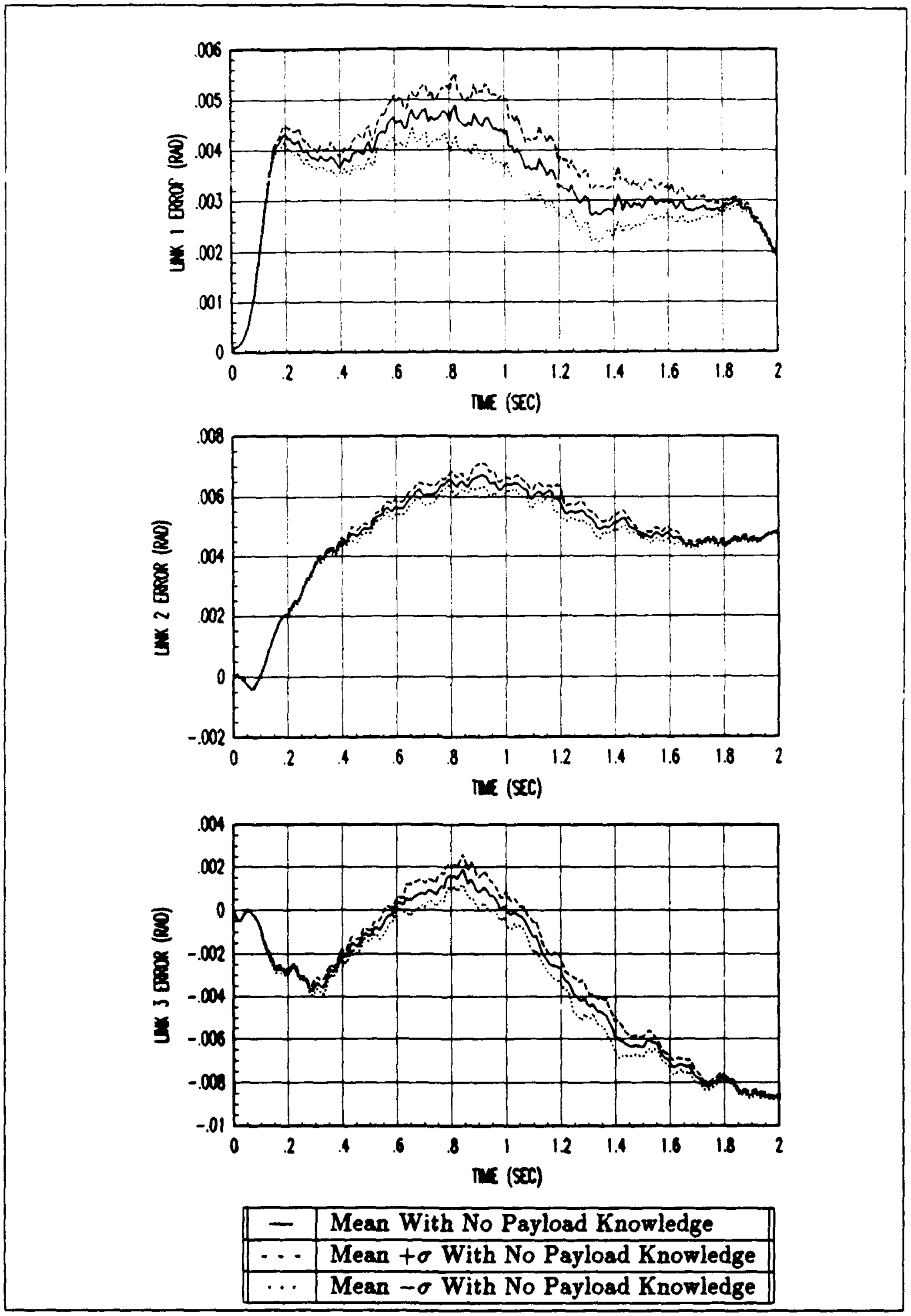


Figure D.16. Payload 2.0 Kg: OL/MMBC Tracking Errors for Trajectory One Using High Gains

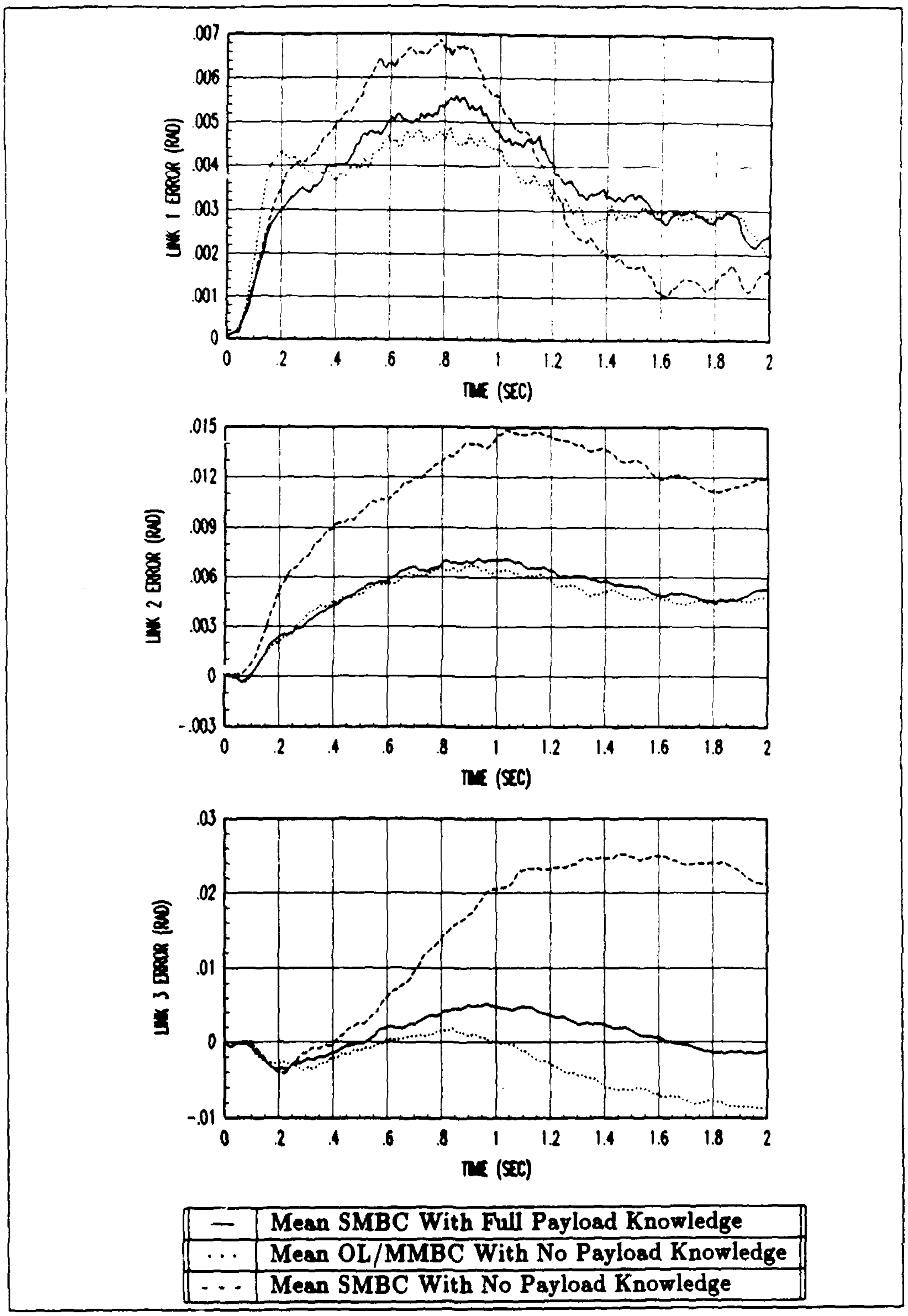


Figure D.17. Payload 2.0 Kg: Tracking Errors for Trajectory One Using High Gains

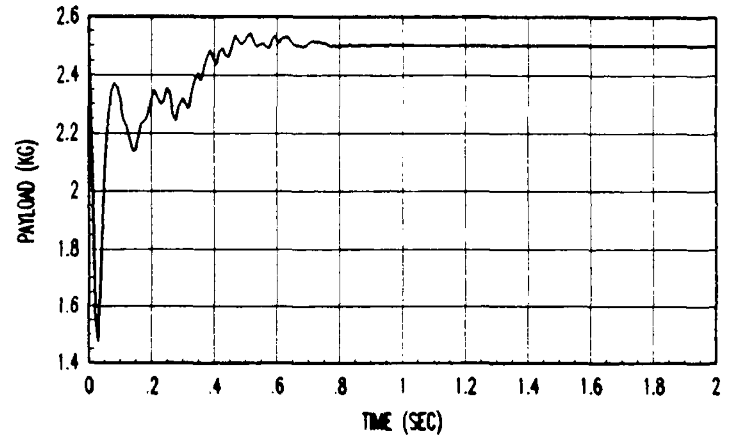


Figure D.18. Payload 2.0 Kg: Mean MMAE Payload Estimate Using High Gains

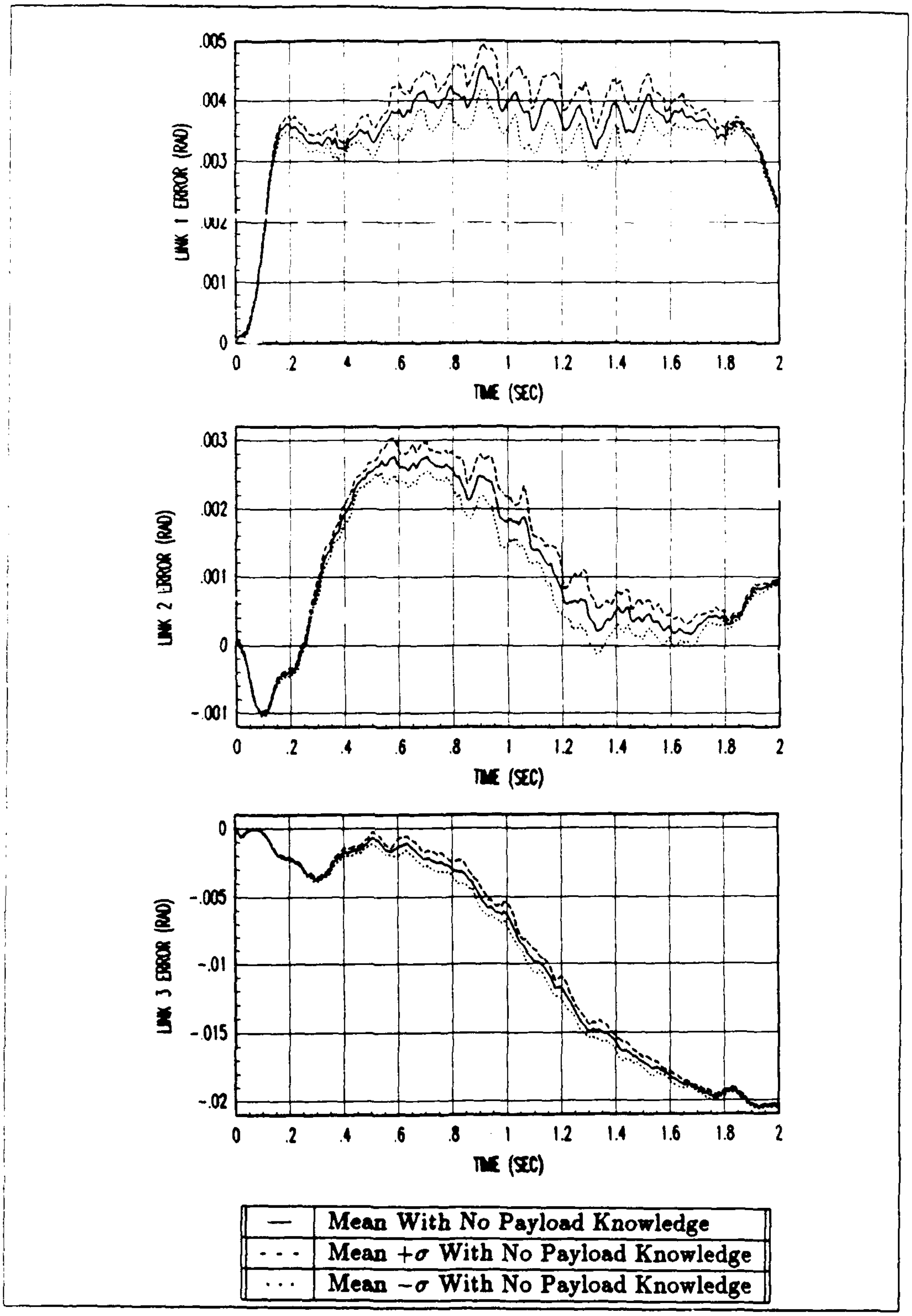


Figure D.19. **Payload 1.0 Kg: OL/M'4BC Tracking Errors for Trajectory One Using High Gains**

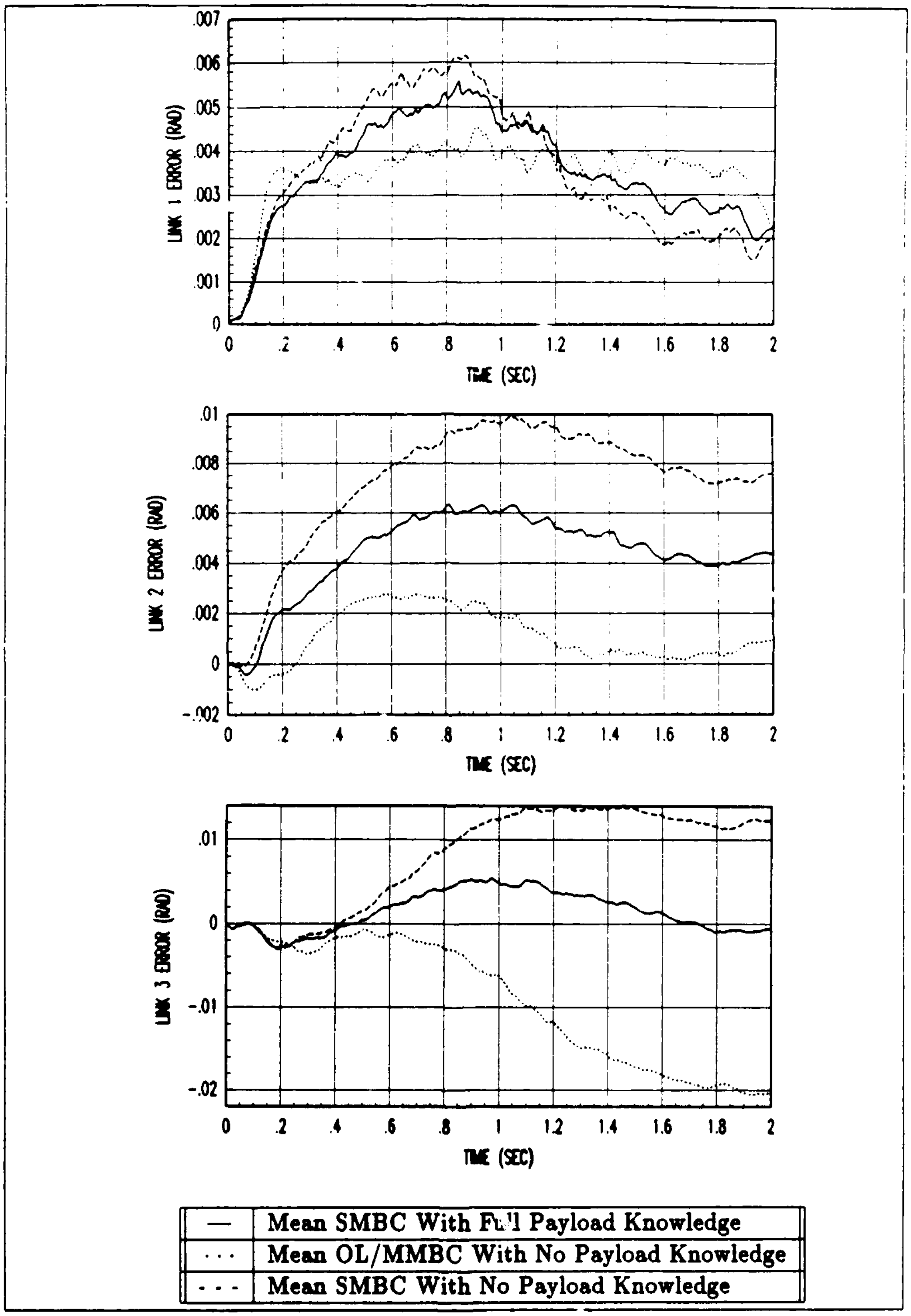


Figure D.20. Payload 1.0 Kg: Tracking Errors for Trajectory One Using High Gains

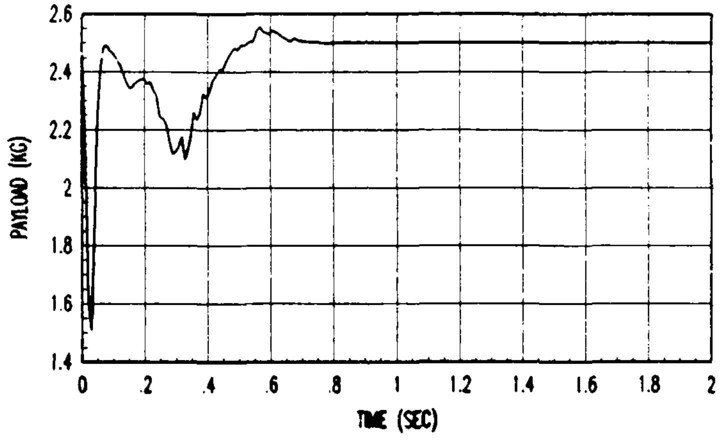


Figure D.21. Payload 1.0 Kg: Mean MMAE Payload Estimate Using High Gains

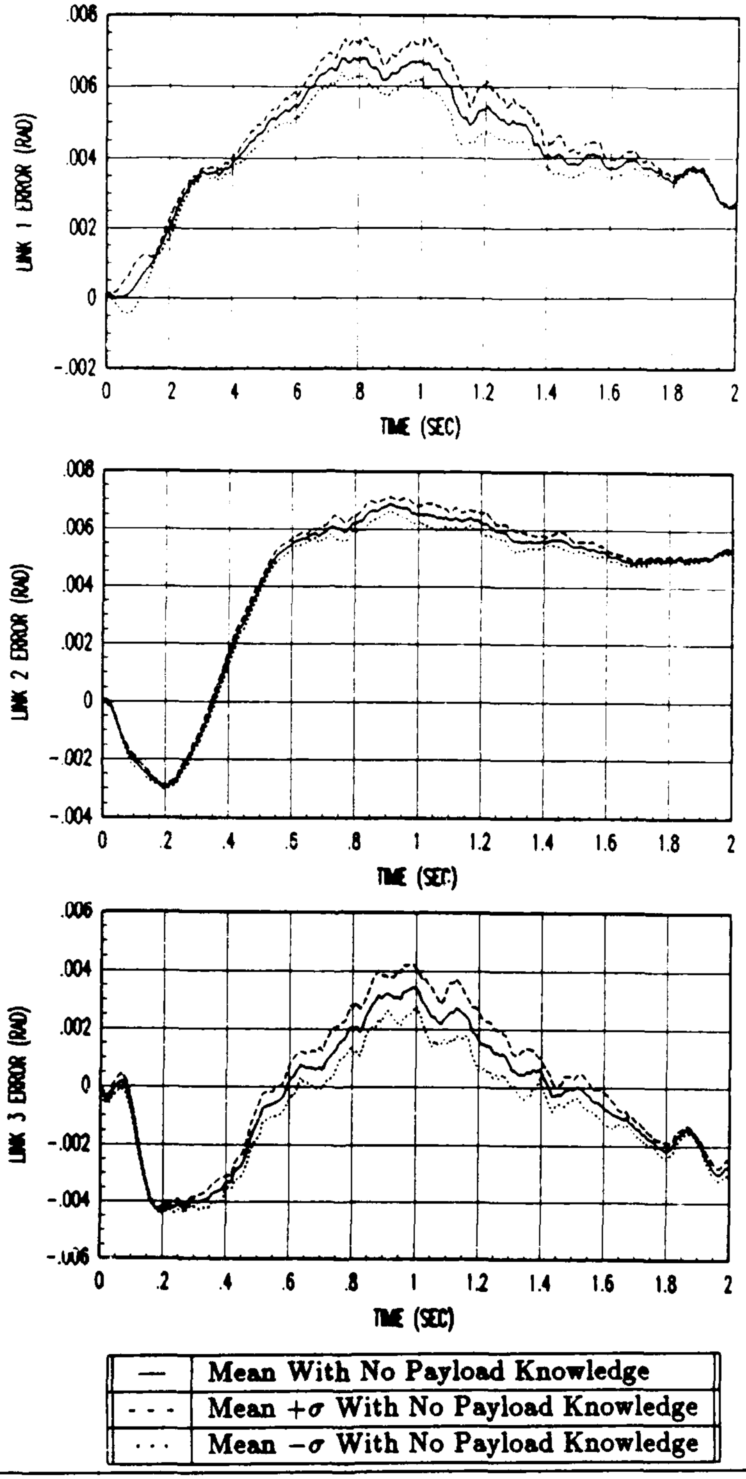


Figure D.22. Payload 0.0 Kg: OL/MMBC Tracking Errors for Trajectory One Using High Gains

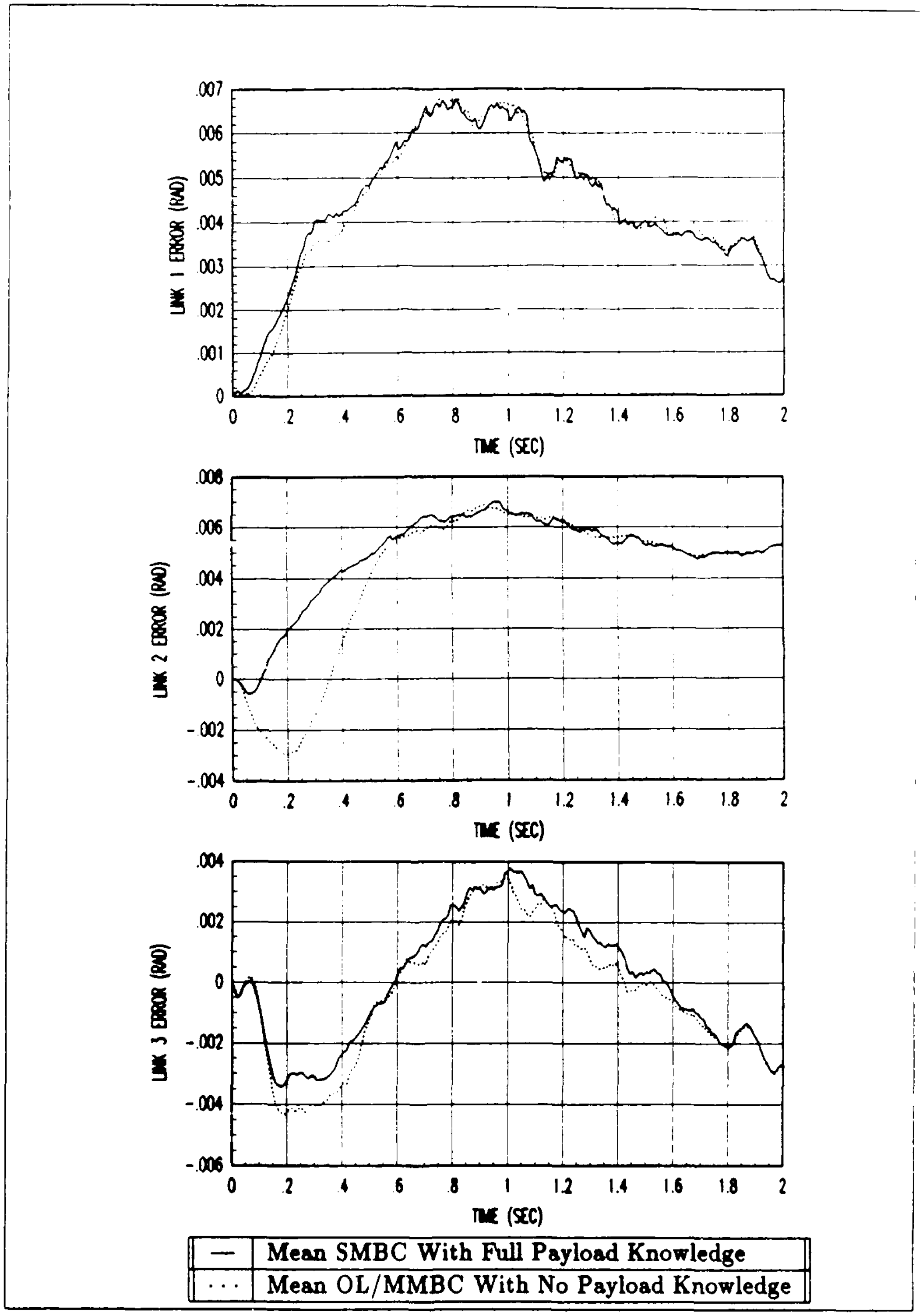


Figure D.23. Payload 0.0 Kg: Tracking Errors for Trajectory One Using High Gains

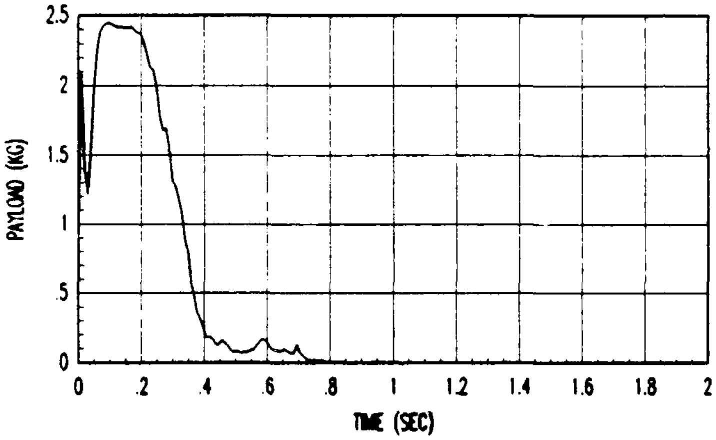


Figure D.24. Payload 0.0 Kg: Mean MMAE Payload Estimate Using High Gains

Bibliography

1. Craig, J.J. et al. "Adaptive Control of Mechanical Manipulators," *Int. Journal of Robotics Research*, Volume 6(2), pp. 16-28, (Summer 1987).
2. deSilva, C.W. and Van Winssen, J. "Least Squares Adaptive Control for Trajectory Following Robots," *Transactions of the ASME*, 109:104–110, June 1987.
3. Digital Inc. *VAXlab Guide to the Laboratory I/O Routines.* December 1987.
4. Dubowsky, S. and DesForges, D.T. "The Application of Model-Referenced Adaptive Control to Robotic Manipulators," *Transactions of the Journal of Dynamic Systems, Measurements, and Control*, 101:225–232, September 1979.
5. EENG 540. "Fundamentals of Robotics," Class Notes, School of Engineering, Air Force Institute of Technology, Air University, September-December 1988.
6. Fu, K.S., Gonzalez, R.C., and Lee, C.S.G. *Robotics: Control, Sensing, Vision and Intelligence*, McGraw-Hill Book Company, New York, 1987.
7. Guo, L. and Angeles, J. "Controller Estimation for the Adaptive Control of Robotic Manipulators," *IEEE Transactions on Robotics and Automation*, Volume 5(3), pp. 315–323, June 1989.
8. Koivo, A.J. and Guo, T-H. "Adaptive Linear Controller for Robotic Manipulators," *IEEE Transactions on Automatic Control*, AC–28(2):162–170, February 1983.
9. Lashlee, R.W. *Moving Bank Multiple Model Adaptive Estimation Applied to Flexible Spacestructure Control.* Master's thesis, School of Engineering, Air Force Institute of Technology, Air University, December 1987.
10. Leahy M.B. "Dynamics Based Control of Vertically Articulated Manipulators," *Proc. of the IEEE Int. Conf. on Robotics and Automation*, pp. 1046–1056, April 1988.
11. — "Dynamics Based Control of Vertically Articulated Manipulators with Variable Payloads," Accepted for publication in *Int. Journal of Robotics Research*, 1989.
12. — "Experimental Analysis of Model-Based Puma Robot Control,". *Proc. IEEE International Symposium on Intelligent Controls*, September 1989.
13. — "ARCADE Users Guide: Version 2.0 Technical Report ARSL–89–4,", Dept. of Electrical and Computer Engineering, Air force Institute of Technology, Air University, WPAFB, OH, July 1989.
14. Leahy, M.B. and Saridis, G.N. "Compensation of Industrial Manipulator Dynamics," *Int. Journal of Robotics Research*, 8(4), August 1989.
15. Leahy, M.B. and Tellman, L.D. "Multiple Model-Based Control of Robotic Manipulators," *Proc. IEEE International Symposium on Intelligent Controls*, September 1989.
16. Lee, C.S.G. and Chung, M.J. "An Adaptive Control Strategy for Mechanical Manipulators," *IEEE Transactions on Automatic Control*, AC–29(9):837–840, September 1984.
17. — "Adaptive Perturbation Control with Feedforward Compensation for Robot Manipulators," *Simulation*, Volume 44, No. 3, pp. 127–136

18. Lim, K.Y. and Eslami, M. "Robust adaptive Controller Designs for Robot Manipulator Systems," *IEEE Journal of Robotics and Automation*, RA-3(3):54-66, February 1987.

19. Maybeck, P.S. *Stochastic Models, Estimation and Control.* Volume 1, Academic Press, Inc., New York: 1979.

20. Maybeck, P.S. *Stochastic Models, Estimation and Control.* Volume 2, Academic Press, Inc., New York: 1982.

21. Maybeck, P.S. August-September 1989. Discussions concerning Multiple Model Adaptive Estimation.

22. Middleton, R.H. and Goodwin, G.C. "Adaptive Computed Torque Control for Rigid Manipulators," *Proc. of the 25th IEEE CDC*, pp. 68-73, 1986.

23. Sablan, S.J. *Software Programs Source Listing for the Open-Loop Multiple Model-Based Control Algorithm Implemented on a PUMA-560 Manipulator.* Internal Report, ARSL-89-13, School of Engineering, Air Force Institute of Technology, Air University, December 1989

24. Seraji, H. "A New Approach to Adaptive Control of Manipulators," *Transactions of the Journal of Dynamic Systems, Measurements, and Control*, 109:193-202, September 1987.

25. Sheldon, S.N. *An Optimal Parameter Discretization Strategy for Multiple Model Adaptive Estimation and Control.* PhD dissertation, School of Engineering, Air Force Institute of Technology, Air University, December 1989.

26. Slotine , J-J. and Li, W. "On the Adpative Control of Robot Manipulators," *Int. Journal of Robotics Research*, Volume 6(3), pp. 49-59, (Fall 1987).

27. —— "Adaptive Manipulator Control: A Case Study". *Proc. of the IEEE Trans. on Automatic Control*, Volume 33, No. 11, pp. 704-9, November 1988.

28. Symbolics Inc. *VAX UNIX MACSYMA Reference Manual.* November 1985.

29. Tarn, T.J. and Bejczy, A.K. *Dynamic Equations for PUMA-560 Robot Arm Technical Report SSM-RL-85-02.* Dept. of Systems Science and Mathematics, Washington University, St. Louis, MO, July 1985.

30. Tellman, L.D. *Multiple Model-Based Robotic Control: Development and Initial Evaluation.* Master's thesis, School of Engineering, Air Force Institute of Technology, Air University, October 1988.

31. Tellman, L.D. and Leahy, M.B. "Multiple Model-Based Control Development and Initial Evaluation,", Internal Report, ARSL-89-1, School of Engineering, Air Force Institute of Technology, Air University, February 1989

32. Unimation Inc. *PUMA Mark II Robot 500 Series Equipment Model.* March 1985.

33. Youcef-Toumni, K. and Kuo A.T.Y. "High Speed Trajectory Control of a Direct-Drive Manipulator," *Proc. of the 26th IEEE CDC*, pp. 2209-2214, December 1987.

Vita

Captain Samuel J. Sablan [redacted] He was selected for the College Senior Engineering Program while attending California State University, Chico. Upon graduation from California State University, Chico in May, 1984, Capt Sablan attended Officer Training School and was commisioned on August 29, 1984. His first assignment was to Edwards AFB, CA, as an avionics flight test engineer. Following the assignment at Edwards AFB, he entered the Air Force Institute of Technology to obtain a Masters Degree in Electrical Engineering in June, 1988.

[redacted]

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION
UNCLASSIFIED

1b. RESTRICTIVE MARKINCS

2a. SECURITY CLASSIFICATION AUTHORITY

2b. DECLASSIFICATION / DOWNGRADING SCHEDULE

3. DISTRIBUTION / AVAILABILITY OF REPORT
Approved for public release;
distribution unlimited

4. PERFORMING ORGANIZATION REPORT NUMBER(S)

5. MONITORING ORGANIZATION REPORT NUMBER(S)

6a. NAME OF PERFORMING ORGANIZATION
School of Engineering

6b. OFFICE SYMBOL *(If applicable)*
AFIT/ENG

7a. NAME OF MONITORING ORGANIZATION

6c. ADDRESS *(City, State, and ZIP Code)*
Air Force Institute of Technology
Wright-Patterson AFB, OH 45433

7b. ADDRESS *(City, State, and ZIP Code)*

8a. NAME OF FUNDING / SPONSORING ORGANIZATION
AAMRL

8b. OFFICE SYMBOL *(If applicable)*
BBA

9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER

8c. ADDRESS *(City, State, and ZIP Code)*
Wright-Patterson AFB, OH 45433

10. SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO	WORK UNIT ACCESSION NO.

11. TITLE *(Include Security Classification)*
see Box 19

12. PERSONAL AUTHOR(S)
Samuel J. Sablan, Capt, USAF

13a. TYPE OF REPORT
MS Thesis

13b. TIME COVERED
FROM ________ TO ______

14. DATE OF REPORT *(Year, Month, Day)*
1989 December

15. PAGE COUNT
157

16. SUPPLEMENTARY NOTATION

17. COSATI CODES

FIELD	GROUP	SUB-GROUP
12	09	
17	07	03

18. SUBJECT TERMS *(Continue on reverse if necessary and identify by block number)*
Robot Control, Parameter Estimation,
Multiple Model Adaptive Estimation

19. ABSTRACT *(Continue on reverse if necessary and identify by block number)*

11) Title: MULTIPLE MODEL ADAPTIVE ESTIMATION TECHNIQUES FOR
ADAPTIVE MODEL-BASED ROBOT CONTROL

Thesis Chairman: Michael B. Leahy Jr, Capt, USAF
Associate Professor of Electrical Engineering

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT
☒ UNCLASSIFIED/UNLIMITED □ SAME AS RPT. □ DTIC USERS

21. ABSTRACT SECURITY CLASSIFICATION
UNCLASSIFIED

22a. NAME OF RESPONSIBLE INDIVIDUAL
Michael B. Leahy Jr, Capt, USAF

22b. TELEPHONE *(Include Area Code)*
(513) 255-9269

22c. OFFICE SYMBOL
AFIT/ENG

Previous editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

Abstract

The use of robotic manipulators for future Air Force applications will require a manipulator capable of emulating the performance of the human arm. To emulate human arm motion, a robot must be capable of adapting quickly and accurately to changes in the environment while maintaining accurate high speed tracking performance. One approach to adaptive robotic control is the use of Multiple Model Adaptive Estimation (MMAE) techniques within a model-based control structure. The MMAE techniques employ a bank of Kalman filters whose models are based on different assumed values of the uncertain parameters. Using this bank of filters, the MMAE provides an estimate of the uncertain parameters. A previous development used a closed-loop form of MMAE with a model-based controller and was called Multiple Model-Based Control (MMBC). Further analysis of the MMBC showed it has limited applications to manipulators whose dynamics and tracking performance depend heavily on the payload. This is not the case for the PUMA-560 manipulator. As a result, a new form of adaptive model based control called Open-Loop Multiple Model-Based Control (OL/MMBC) was developed. The OL/MMBC combines a model-based controller with a MMAE algorithm whose filters are based on an open-loop linearized perturbation model. The OL/MMBC was simulated and experimentally evaluated on a PUMA-560 manipulator. The OL/MMBC algorithm adapted quickly to uncertain payloads and provide payload estimates which resulted in tracking performance significantly better than a model-based controller without knowledge of the payload. Additionally, the tracking performance was comparable to a model-based controller artificially informed of the payload. The experimental evaluations validated the simulation results and show the potential of the OL/MMBC for possible use in future Air Force robotic applications.